**Rauterberg, M., Spinas, P., Strohm, O., Ulich, E. & Waeber, D. (1994).
Benutzerorientierte Software-Entwicklung Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung.
(Mensch - Technik - Organisation. Band 3), vdf Hochschulverlag AG an der ETH Zurich.**

Matthias Rauterberg, Philippe Spinas,
Oliver Strohm, Eberhard Ulich und Daniel Waeber

BENUTZERORIENTIERTE SOFTWARE-ENTWICKLUNG

Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung

Inhaltsverzeichnis

Vorwort V

Lesehinweise..... VII

Einleitung 1

TEIL A: GRUNDLAGEN 5

1 Arbeitspsychologische Grundlagen..... 5
1.1 Soziotechnisches Systemkonzept.....7
1.2 Aufgabengestaltung 11
1.3 Software-Ergonomie 11
1.4 Fazit 26

2 Grundlagen der Benutzerbeteiligung 28
2.1 Gründe für Benutzerbeteiligung 28
2.2 Voraussetzungen für Benutzerbeteiligung 34
2.3 Konzept der Benutzerbeteiligung 39
2.4 Praxisprobleme bei Benutzerbeteiligung..... 47
2.5 Fazit 52

TEIL B: ORGANISATION DER SOFTWARE-ENTWICKLUNG...53

1 Einleitung..... 53

2 Projektorganisation 57
2.1 Die Zieldefinition 57
2.2 Die Projektklassifikation..... 58
2.3 Die Aufbauorganisation..... 62
2.4 Fazit 82

3 Prozessgestaltung 84
3.1 Warum das Phasenmodell nicht genügt 84
3.2 Der Optimierungszyklus 86
3.3 Quadrant-I: Die Analyse- und Bewertungsphase 88
3.4 Quadrant-II: Die Spezifikationsphase 89
3.5 Quadrant-III: Die Realisierungsphase 90
3.6 Quadrant-IV: Die Implementations- und Benutzungsphase 91
3.7 Das iterativ-zyklische Ablaufmodell 92

4 Vorgehensweisen und Methoden in der Praxis 95
4.1 Übersicht 95
4.2 Konzeption eines Beteiligungsmodells 99
4.3 Analyse, Bewertung und Gestaltung bestehender Arbeitssysteme 109
4.4 Vorgehen zur Ermittlung erster Anforderungen an die Software 115
4.5 Der Workshop 119
4.6 Befragung 123
4.7 Das Pflichtenheft 127
4.8 Simulationsmethoden 127
4.9 Formale Darstellungsmethoden 130
4.10 Prototyping 131
4.11 Checklisten 135
4.12 Evaluationstechniken 136
4.13 Codeinspektion 137
4.14 Benutzungs- und Usability-Tests 138
4.15 Schulung 146
4.16 Nachevaluation beim Systemeinsatz 151

5 Fazit 152

TEIL C: FALLBEISPIELE 153

Fall 1: Versicherung 154

1 Ausgangslage 154
2 Projektorganisation 154
3 Projektablauf 155
4 Weiterer Prozessverlauf 157
5 Fazit 158

Fall 2: Bankabteilung KBT 161
1 Ausgangslage 161
2 Projektorganisation 162
3 Projektablauf 163
4 Fazit 165

Fall 3: Bibliothek 167
1 Ausgangslage 167
2 Untersuchung 167
3 Workshop 169

Fall 4: Standardsoftware bei der ADI 171
1 Ausgangslage 171
2 Projektmanagement und Projektablauf 172
3 Fazit 175

Fall 5: Die Einführung von PPS 178
1 Phase I: Die 'eher arbeitsorientierte Idee' 178
2 Phase II: Die 'technikorientierte Realisierung' 179
3 Die Situation nach der Systemeinführung 182
4 Phase III: Die 'eher arbeitsorientierte Korrektur' 184
5 Fazit 185

ANHANG 187

Checkliste 1: 188
Prinzipien und Hinweise für Veränderungsprozesse und die Beteiligung von Benutzern bei der Softwareentwicklung 188

Checkliste 2: 195
Vereinbarungen für einen partnerorientierten demokratischen Gruppenprozess 195

Übersicht 1: 196
Übersicht über Indikatoren für das Scheitern von Softwareprojekten ... 196

Übersicht 2: 198
Übersicht über Hilfsmittel zur Entwurfsunterstützung 198

Übersicht 3: 203
Arbeitsanalyseverfahren 203

Literatur 206

Personenverzeichnis 216

Stichwortverzeichnis 219

Vorwort

Bei Projekten zur Computerunterstützung von Bürotätigkeiten stellt sich die Grundfrage: *Wie* können *Benutzerbedürfnisse bzw. Anforderungen* von Fachabteilungen nach Computerunterstützung von Arbeitsabläufen so *in Software umgesetzt* werden, dass diese den Anforderungen auch wirklich entspricht und dadurch eine *echte Hilfe* bei der Bewältigung der anfallenden Aufgaben darstellt? Allzu oft sind schon Entwicklungsprojekte gescheitert, oder die entwickelten Lösungen haben den Benutzern nicht die erwartete Unterstützung gebracht, weil ihre Bedürfnisse und ihre Aufgaben im Prozess der Softwareentwicklung zu wenig berücksichtigt wurden. Warum? Viele Beispiele zeigen, dass es für den Softwareentwickler sehr schwierig ist, sich in die Lage eines zukünftigen Systembenutzers zu versetzen und dessen Denkvorgänge sowie Handlungen im Arbeitsprozess zu antizipieren. Dies gilt je länger je mehr, denn die Komplexität der Anwendungen steigt immer schneller an, und für den Softwareentwickler wird es zunehmend schwieriger oder gar unmöglich, den fachlichen Hintergrund einer Anwendung zu durchschauen und zu verstehen. Diese Komplexität erfordert eine enge *Zusammenarbeit zwischen Benutzern als Experten für das zu unterstützende Fachgebiet und Entwicklern als Experten für Informatikwerkzeuge*, um optimale, auf die konkrete Situation angepasste Lösungen zu finden. Über die Notwendigkeit einer Beteiligung von Benutzern bei der Softwareentwicklung besteht heute bei Fachleuten verschiedenster Disziplinen weitgehend Einigkeit. Probleme bereitet allerdings häufig die praktische Realisierung.

Das vorliegende Buch ist das Ergebnis eines vierjährigen Projektes zur benutzerorientierten Softwareentwicklung, das vom Bundesministerium für Forschung und Technologie (Projektträger 'Arbeit und Technik, Bonn; Förderkennzeichen 01HK706) finanziell gefördert wurde. Dabei wurden die Entwicklungsprozesse in über 100 Projekten analysiert, Methoden und Vorgehensweisen zur Beteiligung von (End-)Benutzern entwickelt und erprobt sowie Benutzungstests zur Messung der Gebrauchstauglichkeit von interaktiven EDV-Systemen durchgeführt. Die Ergebnisse dieser Analysen, Feldversuche und Benutzungstests wurden aufgearbeitet und in diesem Buch dargestellt. Damit soll Softwareentwicklern (Projektleiter, Analytiker, Programmierer), Benutzern, Organisatoren, Ergonomiebeauftragten, Koordinatoren zwischen EDV- und Fachabteilung und Ausbildnern eine Unter-

stützung bei der Entwicklung benutzerorientierter EDV-Systeme angeboten werden.

Obwohl bereits vor mehr als zehn Jahren eine Reihe von – zum Teil heute noch aktuellen Büchern – zum Thema der Benutzerbeteiligung erschienen sind, erscheint es sinnvoll, dieses Buch zu veröffentlichen. Die drei folgenden Gründe haben uns dazu bewogen:

(1) Es wurden in den letzten Jahren weltweit viele neue Methoden, Techniken und Formen für Benutzerbeteiligungen entwickelt und erprobt.

(2) Dieses Buch zeichnet sich durch eine integrative Sichtweise aus – die Integration von arbeitspsychologischen und softwaretechnischen Aspekten.

(3) Da die meisten Bücher über moderne Methoden englischsprachig sind, erschliesst dieses deutschsprachige Buch den Zugang zu dieser Literatur und gibt eine erste Orientierung.

Neben der Vermittlung (arbeits-)psychologischer Grundlagen werden Konzepte der Benutzerbeteiligung vorgestellt, Hinweise und Ideen für die praktische Realisierung gegeben sowie mögliche Probleme und Lösungsansätze aufgezeigt. Fallbeispiele, Checklisten und Übersichten unterstützen die Umsetzung der Anregungen in das Vorgehen bei konkreten Projekten. Wenn auch noch nicht alle Probleme im Detail gelöst sein mögen und bereits vorhandene Lösungsmöglichkeiten aufgrund des begrenzten Seitenumfangs zum Teil nicht weiter ausgeführt werden konnten, so gibt dieses Buch dennoch eine Fülle an Anregungen und Hinweisen auf weiterführende Literatur.

An dieser Stelle sei allen Beschäftigten in den Betrieben gedankt, durch deren Kooperationsbereitschaft im Laufe der letzten vier Jahre viele wertvolle Erkenntnisse und Erfahrungen in dieses Buch eingeflossen sind. Insbesondere gilt unser Dank dem Geschäftsführer Herrn Dr. Karl Schlagenhauf und dem Produktmanager Herrn Raimund Mollenhauer der ADI Software GmbH in Karlsruhe für die fruchtbare Zusammenarbeit.

Zürich, im Juli 1994

Matthias Rauterberg, Philipp Spinas, Oliver Strohm, Eberhard Ulich und Daniel Waeber

Lesehinweise

Für dieses Buch gibt es verschiedene Leseweisen, welche sich nach den unterschiedlichen Interessen der Leser ausrichten. Wir bieten daher die folgenden Leseweisen an:

Als Nachschlagewerk – Zuerst im Stichwort- oder Inhaltsverzeichnis die relevanten Verweise heraussuchen und dann an den entsprechenden Textstellen weiterlesen.

Falls bei den folgenden Lesehinweisen einige Begriffe an der jeweiligen Stelle nicht ausreichend erklärt sein sollten, so kann dies daran liegen, dass diese Begriffe im Grundlagenteil A eingeführt wurden.

Hinweise auf Methoden – Im Teil B ist im Kapitel 4 eine weitgehend vollständige Auflistung und Darstellung der verschiedenen, zur Zeit bekannten Methoden zur Benutzerbeteiligung gegeben. Zur Vertiefung bei der einen oder anderen Methode kann es unumgänglich sein, auf die weiterführende Literatur zurückzugreifen.

Fragen zur Projektleitung und zum Projektaufbau – Im Teil B sind in Kapitel 2 verschiedene Ansätze zur Projektorganisation mit Benutzerbeteiligung dargestellt. Falls einige Begriffe nicht ausreichend erklärt sein sollten, so kann dies daran liegen, dass diese Begriffe im Grundlagenteil A eingeführt wurden.

Fragen an den Projektablauf und die Prozessgestaltung – Im Teil B ist in Kapitel 3 ein iterativ-zyklisches Ablaufmodell vorgestellt, welches die Zuordnung der Methoden zum Benutzereinbezug an den verschiedenen Stellen im Entwicklungsprozess vorsieht.

Für den Einsatz in der Lehre – Anhand der dargestellten Fälle im Teil C lassen sich verschiedene Aspekte für Benutzerbeteiligung herausarbeiten. Je nach Fragestellung werden dann entsprechende Vertiefungen in den Teilen A und B bzw. der weiterführenden Literatur notwendig.

Einleitung

Mit der fortschreitenden Automatisierung von Arbeitsprozessen – man denke etwa an Schlagworte wie Bürokommunikation, Büroautomation oder CIM, CAD, CAM im Produktionsbereich – werden verschiedene Ziele verfolgt. Beispielhaft seien hier die Steigerung von Effizienz und Produktivität, die Verbesserung von Dienstleistungen, die Erhöhung von Wirtschaftlichkeit und Konkurrenzfähigkeit, die Verkürzung der Durchlaufzeiten von Werkstücken und Dokumenten, die Verbesserung von Arbeitsbedingungen, die Erhöhung der Auskunftsbereitschaft durch Integration von Informationen und verbesserten Zugriffsmöglichkeiten usw. genannt. Diese Ziele werden allerdings von den unterschiedlichen Interessengruppen im Betrieb – z.B. Management, Systementwickler, Organisatoren und Benutzer – vielfach unterschiedlich gewichtet, was zu Interessenkonflikten führen kann. Immer häufiger stellt sich deshalb bei Vorhaben zur Computerunterstützung von Arbeitstätigkeiten die folgende Grundfrage: Wie können Anforderungen von Fachabteilungen bzw. Bedürfnisse von Benutzern nach Computerunterstützung von Arbeitsabläufen so in Software umgesetzt werden, dass diese den Anforderungen auch wirklich entspricht und dadurch eine echte Hilfe bei der Bewältigung der anfallenden Aufgaben darstellt?

In den letzten Jahren liessen sich in vielen Softwareentwicklungsprojekten grosse Probleme feststellen, die manchmal zum Scheitern eines Projektes oder gar zum Konkurs einer Firma führten. Die Ursachen können vielfältiger Natur sein, denn für die Akzeptanz eines neuen Systems durch die Benutzer spielt die optimale Gestaltung folgender, unterschiedlicher Aspekte eine Rolle:

(1) Arbeitsteilung und -organisation,

(2) Mensch-Computer-Funktionsteilung,

(3) Benutzungsoberfläche (Softwareergonomie),

(4) Ergonomie von Hardware sowie Arbeitsplatz und -umgebung,

(5) Entwicklungs- und Einführungsprozess, Benutzerbeteiligung.

In diesem Buch befassen wir uns *zur Hauptsache* mit den Möglichkeiten der Benutzerbeteiligung im Prozess der Softwareentwicklung (siehe auch Mambrey und Oppermann 1983, Mumford und Welter 1984, Mambrey,

Oppermann und Tepper 1986, Peschke 1986, Kissler 1988, Koslowski 1988 und Ruch, Spinas und Ulich 1989). In bezug auf die anderen Aspekte verweisen wir auf die entsprechende Fachliteratur. Von besonderer Bedeutung ist die Gestaltung der Arbeitsteilung und -organisation; sie hat vor bzw. zumindest gleichzeitig mit dem Einkauf oder der Entwicklung technischer Systeme zu erfolgen!

Bei den erwähnten Problemen handelte es sich aber meist weniger um technische Schwierigkeiten. Vielmehr ist es die fachliche Komplexität und Dynamik heutiger Anwendungen, welche die Softwareentwickler vor Probleme stellt. Ein EDV-Spezialist ist heutzutage häufig nicht mehr in der Lage, die fachliche Seite einer Anwendung hinreichend zu durchschauen und zu bewältigen. Deshalb ist eine enge Zusammenarbeit von Benutzern als Experten für das zu unterstützende Fachgebiet und Softwareentwicklern als Experten für Informatikwerkzeuge unabdingbar. In manchen Fällen kann auch zusätzlich ein 'neutraler' Prozessmoderator notwendig sein. Im Rahmen partizipativer Softwareentwicklung sind verschiedene Komponenten und Aspekte zu unterscheiden, die in diesem Buch ausführlich dargestellt werden: Rahmenbedingungen, Projektmanagement, Aufbau- und Ablauforganisation, Information, Ausbildung, Partizipation sowie verschiedene Methoden zur Unterstützung von Benutzerbeteiligung.

Entgegen häufig vorgebrachten Argumenten – wie z.B. die Benutzer wollen oder können gar nicht an einer Entwicklung beteiligt werden – ist die Bereitschaft zur Zusammenarbeit von Entwicklern und Benutzern nach den Ergebnissen unserer Untersuchungen viel höher als gemeinhin angenommen. Ein Projektleiter formulierte es – stellvertretend für viele andere – wie folgt: "Ohne die aktive Mitarbeit der Benutzer könnten wir heute keine Anwendung mehr realisieren. Der *erforderliche Zeitaufwand* ist im Verhältnis zu den *erzielten Vorteilen gering*."

Mit dem Einbezug von Benutzern in den Entwicklungsprozess werden primär folgende Ziele verfolgt:

Innovation Es werden innovativere Lösungen entwickelt, in welche neben dem technischen Wissen der Entwickler auch das Fachwissen der Benutzer einfliesst.

Identifikation Die Entwickler wie auch die Benutzer können sich mit der Entwicklungsarbeit sowie auch mit der Lösung besser identifizieren (Akzeptanz!)

Zur Erreichung dieser Ziele müssen folgende Grundbedingungen erfüllt sein:

Integration Aufgabenspezifische, benutzerspezifische und technische Anforderungen müssen *integrativ* analysiert und in der Umsetzung berücksichtigt werden.

Interaktion Entwickler und Benutzer müssen die Möglichkeit haben, direkt miteinander zu interagieren, damit sich gegenseitiges Verständnis entwickeln kann.

Iteration Der Entwicklungsprozess muss so angelegt sein, dass innerhalb einzelner Phasen wie auch über Phasen hinweg iteriert werden kann. Nur so können Veränderungen in den Anforderungen aufgefangen, getroffene Entscheidungen überprüft, Gestaltungsvorschläge, Konzepte evaluiert und gegebenenfalls frühzeitig korrigiert werden.

Inzwischen haben auch namhafte Hersteller sowie Anwender mit eigener Entwicklungsabteilung die Vorteile des Einbezugs von Endbenutzern erkannt. So wirbt z.B. ein Grossproduzent von Standardsoftware in einem doppelseitigen Inserat mit dem Argument der Benutzerbeteiligung für sein Produkt: "An ... haben 300 Leute mitgearbeitet, die keine Ahnung vom Computer haben. ... Das Usability Lab, ein Speziallabor, in dem wir alle unsere neuen Programme sowohl von PC-Freaks als auch von blutigen Laien testen lassen ... Dadurch konnten wir aus jedem Fehler lernen und schwierige Aufgaben ... für jedermann verständlich machen." Ferner ist uns bekannt, dass zwei Schweizer Banken die Beteiligung von Benutzern bezüglich Funktion, Kompetenzen und Verantwortung in Projekthandbüchern verbindlich beschreiben und ausserdem ein Usability Lab bzw. eine Fachstelle für Softwareergonomie eingerichtet haben. In ähnlicher Weise wird der Benutzereinbezug auch bei verschiedenen Unternehmen (z.B. Versicherungen) in Projekthandbüchern verbindlich festgelegt. Diese Beispiele zeigen, dass Benutzerbeteiligung kein theoretisches Konzept mehr, sondern inzwischen praktische Realität ist!

Dieses Buch ist zur besseren Orientierung in drei Teile gegliedert: In *Teil A* werden die wichtigsten *Grundlagen* für aufgaben- und benutzerorientierte Softwareentwicklung erläutert. *Teil B* beschreibt das *praktische Vorgehen*. In *Teil C* sind dann *konkrete Fallbeispiele* beschrieben. Die Fallbeispiele dienen der Veranschaulichung und können auch in der Ausbildung Einsatz finden. Im *Anhang* finden sich *Checklisten* und *Übersichten.* Die Checklisten erlauben direkten Zugriff bei Fragen und Problemen der Alltagsarbeit.

An dieser Stelle scheint uns auch der Hinweis wichtig, was dieses Buch *nicht* beabsichtigt: Es wird *kein* Grundlagenwissen hinsichtlich Organisationsentwicklung und Arbeitsstrukturierung, Softwareentwicklung und Betriebswirtschaftslehre vermittelt. Dieses Buch ersetzt somit kein anderes Buch zu den genannten Bereichen. Das Wissen über die anderen Bereiche wird zum Teil vorausgesetzt bzw. muss ergänzend zur Kenntnis genommen werden. Entsprechende Literaturhinweise finden sich an den jeweiligen Textstellen. Um Missverständnissen vorzubeugen, seien hier einige grundlegende Begriffe kurz erläutert:

(1) Unter Partizipation bzw. Beteiligung oder Einbezug von Benutzern ist deren Mitwirkung an der Planung und Entwicklung eines Systems und die Einflussnahme auf Entscheidungen zu verstehen.

(2) Als Benutzer oder Endbenutzer werden diejenigen Personen (Sachbearbeiter, Sekretärin, Manager usw.) bezeichnet, die das System nach abgeschlossener Entwicklung im Rahmen ihrer täglichen Arbeit benutzen. Im Gegensatz dazu ist mit Käufer der Betreiber oder Anwender gemeint (Organisation, Abteilung oder deren Repräsentanten).

(3) Bezeichnungen wie Entwickler, Designer, Programmierer werden in umfassendem Sinne synonym verwendet für Personen, die EDV-seitig an der Konzeption und Realisierung eines Systems beteiligt sind. In einzelnen Fällen wird eine bestimmte Gruppe – z.B. Projektleiter, Analytiker – besonders hervorgehoben.

(4) In diesem Buch wird fast ausschliesslich die männliche Form (Benutzer, Entwickler usw.) verwendet. Wir gehen aber davon aus, dass beide Geschlechter gemeint sein können (BenutzerIn, EntwicklerIn usw.).

Abschliessend sei noch auf folgende wichtige Unterscheidung hingewiesen: Wir haben es bei Softwareentwicklungsprojekten sowohl mit der Aufbau- und Ablauforganisation eines Unternehmens bzw. der betroffenen Fachabteilungen zu tun als auch mit der Aufbau- und Ablauforganisation des Softwareentwicklungsprojektes selbst. Diese Organisationsformen können unabhängig voneinander sein, aber auch weitgehend identisch. Wir haben diesem Umstand Rechnung getragen, indem wir die arbeitspsychologischen Kriterien für humane Arbeit, welche primär auf die Aufbau- und Ablauforganisation eines Unternehmens ausgerichtet sind, auch auf die Aufbau- und Ablauforganisation der Softwareentwicklung selbst übertragen und entsprechend angepasst haben.

TEIL A: GRUNDLAGEN

1 Arbeitspsychologische Grundlagen

Damit Arbeitstätigkeiten und Arbeitsstrukturen menschengerecht gestaltet werden können, müssen bestimmte Zielvorgaben für die Bewertung bekannt sein. Die Arbeitspsychologie bewertet Arbeit dann als menschengerecht, wenn sie die folgenden vier Kriterien erfüllt (Ulich 1994):

(1) Schädigungsfreiheit,

(2) Beeinträchtigungslosigkeit,

(3) Persönlichkeitsförderlichkeit,

(4) Zumutbarkeit.

Das Kriterium der Schädigungsfreiheit meint, dass Arbeit keine physischen und psychophysischen *Schädigungen* beim Arbeitenden hervorrufen darf. Schädigungen zeichnen sich vor allem dadurch aus, dass sie

(1) objektiv feststellbar sind,

(2) in den normalen Erholungszeiten nicht zurückgebildet werden können und

(3) meistens der Behandlung bedürfen.

Schädigungen durch Arbeit am Computer können beispielsweise in Form degenerativ-rheumatischer Erkrankungen infolge ausschliesslich sitzender Tätigkeit auf ergonomisch schlechten Stühlen bei ganztägiger, intensiver Bildschirmarbeit auftreten.

Im Gegensatz zu den Schädigungen sind *Beeinträchtigungen* des psychosozialen Wohlbefindens weniger gut objektivierbar und können kaum eindeutig auf bestimmte Arbeitsbedingungen zurückgeführt werden. Als psychosoziale Beeinträchtigungen bei Bildschirmarbeit gelten zum Beispiel depressive Verstimmungen als Folge sozialer Isolation am Arbeitsplatz (Einzelarbeit) oder Einschränkungen im Sozialleben bei langjähriger Schichtarbeit in der EDV-Überwachung. Es muss beachtet werden, dass aus Beeinträchtigungen, die über längere Zeit bestehen bleiben, psychosomatische Schädigungen entstehen können.

Aus den genannten Beispielen zu möglichen Schädigungen und Beeinträchtigungen bei der Bildschirmarbeit geht bereits hervor, dass hier nicht die Technik als solche verantwortlich gemacht werden kann, sondern dass deren Einsatzform – ganztägige Bildschirmarbeit, isolierter Arbeitsplatz – bestimmend wirkt. Es ist daher sicherlich eine Aufgabe der Softwareentwicklung, solche Einsatzformen durch entsprechende Gestaltungsmassnahmen in der Software zu verhindern oder wenigstens nicht zwingend festzuschreiben.

Das Kriterium der *Persönlichkeitsförderlichkeit* geht davon aus, dass sich die Persönlichkeitsentwicklung des erwachsenen Menschen vor allem in der Auseinandersetzung mit seiner Arbeitstätigkeit vollzieht. Obwohl nicht abschliessend beantwortet werden kann, welche Persönlichkeitsdimensionen in der Arbeit besonders gefördert werden können, weisen bisherige Untersuchungen klar darauf hin, dass die Ausprägung

(1) der kognitiven Kompetenz,

(2) der sozialen Kompetenz,

(3) des Selbstkonzepts und

(4) der Leistungsmotivation in direktem Zusammenhang mit Merkmalen der Arbeit zu sehen sind.

Als wichtigste Merkmale der Arbeit wirken sich dabei vor allem die Arbeitsinhalte und die Anforderungen, die an die Qualifikationen der Arbeitenden gestellt werden, auf die Persönlichkeitsentwicklung aus. So erhalten

und steigern beispielsweise abwechslungsreiche, herausfordernde Aufgaben die geistige Beweglichkeit.

Zentrale Voraussetzung für die persönlichkeitsförderliche Wirkung von Arbeitstätigkeiten ist deren vollständige Struktur. *Vollständigkeit* ist nach Ulich (1994) gegeben, wenn

(1) Ziele selbständig gesetzt und in übergeordnete Ziele eingebettet werden können,

(2) selbständige Handlungsvorbereitung im Sinne der Wahrnehmung von Planungsfunktionen möglich ist,

(3) die Mittel einschliesslich der erforderlichen Interaktionen zur adäquaten Zielerreichung ausgewählt werden können,

(4) während der Ausführung Rückmeldungen möglich sind, damit allenfalls korrigiert werden kann und

(5) das Ergebnis auf Übereinstimmung mit den gesetzten Zielen überprüft werden kann.

Die Vollständigkeit der Arbeitstätigkeit ist eng verknüpft mit der Arbeitsaufgabe. Sie bildet deshalb den Ansatzpunkt für die Gestaltung menschengerechter Arbeit.

Die *Zumutbarkeit* ist ein gruppenspezifisches Kriterium, welches stark von gesellschaftlichen Normen und Werten bestimmt ist. Ob eine Arbeit für eine bestimmte Person zumutbar ist oder nicht, hängt in erster Linie von den Wertvorstellungen, Bedürfnissen und Ansprüchen der direkt Betroffenen ab. In diesem Sinne kann es auf der Ebene jedes der hier genannten Kriterien angewendet werden.

1.1 SOZIOTECHNISCHES SYSTEMKONZEPT

In der Praxis ist oft zu beobachten, dass die Entwicklung computerunterstützter Arbeitssysteme 'von innen nach aussen' erfolgt. Das heisst: Man geht von der Hardware aus, entwirft Software dafür, setzt als letztes noch eine Benutzungsoberfläche darauf und fragt sich ganz am Schluss, wozu ein solches System benutzt werden kann. Ein derartiges Vorgehen verlangt nicht nur vielfältige nachträgliche Korrekturen, sondern führt möglicherweise auch zu Veränderungen von Tätigkeits- und Organisationsstrukturen, die ursprünglich gar nicht beabsichtigt waren. Wie für die industrielle Fertigung gilt demzufolge auch für computerunterstützte Büroarbeit, dass die

Systeme 'von aussen nach innen' entwickelt werden sollten. Das heisst: Zunächst muss das soziotechnische System, in das das Computersystem eingebettet werden soll, analysiert bzw. konzipiert werden, erst daran anschliessend wird die geeignete Hard- und Software beschafft bzw. angepasst oder neu entwickelt.

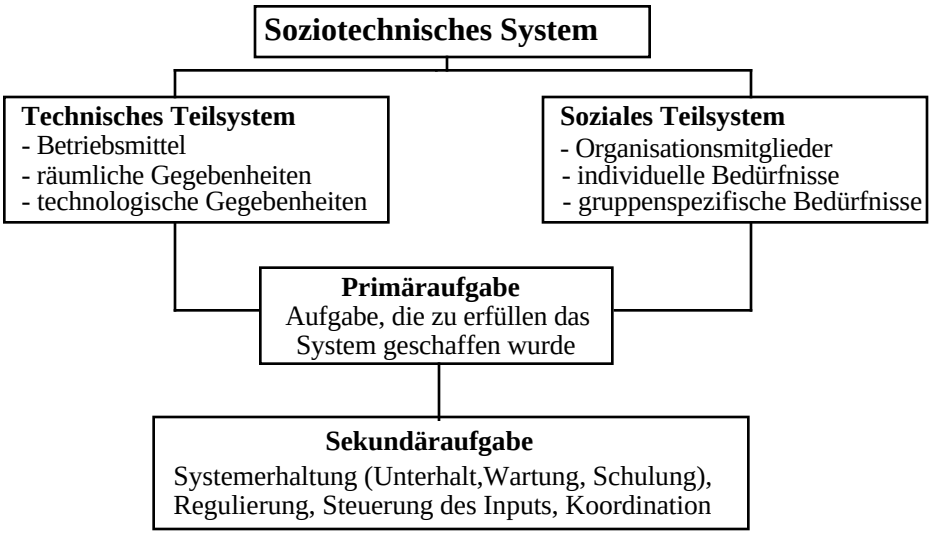


Abbildung 1: Das soziotechnische Systemkonzept.

Jeder Betrieb, jede Organisation muss als soziotechnisches System betrachtet werden, das aus zwei Teilsystemen besteht (siehe Abbildung 1). Von entscheidender Bedeutung ist dabei:

(1) Das technische und das soziale Teilsystem 'funktionieren' nach unterschiedlichen Regeln.

(2) Nur durch eine *gemeinsame Optimierung* des technischen *und* sozialen Teilsystems kann das *ganze* System optimal gestaltet werden.

Um optimale Lösungen zu erreichen, kann man also nicht jedes Teilsystem getrennt für sich optimieren oder ein Teilsystem an das andere anpassen, wie dies häufig zu beobachten ist. Das Konzept der soziotechnischen Systemgestaltung postuliert *explizit* die Notwendigkeit, den Einsatz von Technologie und die Veränderung der Organisation *gemeinsam* zu optimieren.

Das bedeutet nun aber keinesfalls, dass bei Innovationen immer der gesamte Betrieb oder eine ganze Abteilung reorganisiert werden muss. Wesentlich ist jedoch, dass für derartige Vorhaben eine soziotechnische Einheit des Betriebes ausgewählt wird, die ein gegenüber anderen Betriebsbereichen klar abgegrenztes Aufgabengebiet bearbeitet.

Ausgangspunkt für die verknüpfte Optimierung einer soziotechnischen Betriebseinheit ist deren Primäraufgabe und die Berücksichtigung einer Reihe von *Prinzipien* zur soziotechnischen Systemgestaltung (vgl. Ulich, Baitsch und Alioth 1987, S. 24f.):

(1) *Bildung von relativ unabhängigen Organisationseinheiten*
Den relativ unabhängigen Organisationseinheiten sind – als Mehrpersonenstellen – ganzheitliche Aufgaben zu übertragen, so dass sie aufgrund ihrer Unabhängigkeit und der Ganzheitlichkeit der Aufgaben in der Lage sind, Schwankungen und Störungen am Entstehungsort aufzufangen und selbst zu regulieren.

(2) *Aufgabenzusammenhang innerhalb einer Organisationseinheit*
Die verschiedenen Teilaufgaben innerhalb der Organisationseinheit müssen einen inhaltlichen Zusammenhang aufweisen, damit das Bewusstsein einer gemeinsamen Aufgabe entsteht und erhalten werden kann.

(3) *Einheit von Produkt und Organisation*
Aufbau- und Ablaufstrukturen müssen so gestaltet sein, dass Arbeitsergebnisse sowohl qualitativ als auch quantitativ Organisationseinheiten zugeordnet werden können.

Diese drei Prinzipien ermöglichen einerseits die Selbstregulation von Schwankungen und Störungen, welche an ihrem Entstehungsort aufgefangen werden können und somit nicht unkontrolliert auf eine andere Organisationseinheit übertragen werden. Andererseits bedingen sie die Grenzregulation durch den Vorgesetzten: Die Hauptaufgabe des Vorgesetzten besteht darin, die Beziehungen zwischen den verschiedenen Organisationseinheiten sicherzustellen und die Selbstregulation und Unabhängigkeit der Organisationseinheiten zu gewährleisten.

Diese Prinzipien verhindern unter anderem das Entstehen von technischen 'Sachzwängen', die sich üblicherweise dann ergeben, wenn technische Systeme ohne Berücksichtigung von Organisationsanforderungen konzipiert werden.

Tabelle 1: Merkmale der Aufgabengestaltung (nach Ulich 1994, S. 161).

Gestaltungsmerkmal	**Ziel / Absicht Vorteil / Wirkung**	**Realisierung durch ...**
Ganzheitlichkeit	• Mitarbeiter erkennen Bedeutung und Stellenwert ihrer Tätigkeit • Mitarbeiter erhalten Rückmeldung über den eigenen Arbeitsfortschritt aus der Tätigkeit selbst	... umfassende Aufgaben mit der Möglichkeit, Ergebnisse der eigenen Tätigkeit auf Übereinstimmung mit gestellten Anforderungen zu prüfen
Anforderungsvielfalt	• Unterschiedliche Fähigkeiten, Kenntnisse und Fertigkeiten können eingesetzt werden • Einseitige Beanspruchungen können vermieden werden	... Aufgaben mit planenden, ausführenden und kontrollierenden Elementen bzw. unterschiedlichen Anforderungen an Körperfunktionen und Sinnesorgane
Möglichkeiten der sozialen Interaktion	• Schwierigeiten können gemeinsam bewältigt werden • Gegenseitige Unterstützung hilft Belastungen besser ertragen	... Aufgaben, deren Bewältigung Kooperation nahelegt oder voraussetzt
Autonomie	• Stärkt Selbstwertgefühl und Bereitschaft zur Übernahme von Verantwortung • Vermittelt die Erfahrung, nicht einfluss- und bedeutungslos zu sein	... Aufgaben mit Dispositions- und Entscheidungsmöglichkeiten
Lern- und Entwicklungsmöglichkeiten	• Allgemeine geistige Flexibilität bleibt erhalten • Berufliche Qualifikationen werden erhalten und weiterentwickelt	... problemhaltige Aufgaben, zu deren Bewältigung vorhandene Qualifikationen erweitert bzw. neue Qualifikationen erworben werden müssen
Zeitelastizität und stressfreie Regulierbarkeit	• Wirkt unangemessener Arbeitsverdichtung entgegen • Schafft Freiräume für stressfreies Nachdenken und selbstgewählte Interaktionen	... Schaffen von Zeitpuffern bei der Festlegung von Vorgabezeiten
Sinnhaftigkeit	• Vermittelt das Gefühl, an der Erstellung gesellschaftlich nützlicher Produkte beteiligt zu sein • Gibt Sicherheit der Übereinstimmung individueller und gesellschaftlicher Interessen	... Produkte, deren gesellschaftlicher Nutzen nicht in Frage gestellt wird ... Produkte und Produktionsprozesse, deren ökologische Unbedenklichkeit überprüft und sichergestellt werden kann

1.2 AUFGABENGESTALTUNG

Arbeitsaufgaben stellen die Schnittstelle zwischen der Organisation und dem Individuum dar und bestimmen, welche Ziele eine Person oder Gruppe im Rahmen eines Gesamtauftrages erreichen soll.

Für die Erfüllung der Aufgaben stehen bestimmte Arbeitsmittel oder Werkzeuge zur Verfügung. Der Computer als technisches Unterstützungsmittel bei der Aufgabenbearbeitung verkörpert in diesem Sinne also ein Werkzeug.

Der zentrale Ansatzpunkt für die Gestaltung von Aufgaben bezieht sich auf deren *Motivationsgehalt*. Eine Aufgabe sollte so gestaltet sein, dass der Mensch sie als Herausforderung erlebt und Freude an ihrer Bewältigung hat. Eine solche innere Motivation aus der Aufgabe heraus entsteht dann, wenn bei der Gestaltung von Arbeitsaufgaben die folgenden sieben Gestaltungsmerkmale berücksichtigt werden (siehe Tabelle 1).

Aufgaben, die nach den Merkmalen 'Ganzheitlichkeit', 'Anforderungsvielfalt', 'Möglichkeiten der sozialen Interaktion', 'Autonomie', 'Lern- und Entwicklungsmöglichkeiten', 'Zeitelastizität' sowie 'Sinnhaftigkeit' gestaltet sind, fördern den Menschen in seiner sozialen und kognitiven Kompetenz, optimieren seine Beanspruchung, steigern sein Selbstwertgefühl und berücksichtigen seine Entwicklungsfähigkeit (Ulich 1994). Aufgaben werden vor allem auch dann als interessant und motivierend erlebt, wenn in ihnen planende, ausführende und kontrollierende Funktionen integriert sind. Dies kann zum Beispiel durch die Sachbearbeitung nach dem 'Fall-Prinzip' – ein Angestellter erfüllt alle im Zusammenhang mit einem Geschäftsfall zu erledigenden Teilaufgaben – realisiert werden.

1.3 SOFTWARE-ERGONOMIE

Wenn man den Menschen mit dem Computer vergleicht, so kann man feststellen, dass es sowohl für den Menschen als auch für den Computer spezifische Stärken und Schwächen gibt (siehe Tabelle 2). Der Computer als Arbeitsmittel dient primär dazu, den Benutzer bei seinen Arbeiten zu unterstützen und von unzumutbaren, sich ständig wiederholenden, z.B. hochgradig routinisierten Aufgaben zu entlasten. Dies bedeutet jedoch nicht, dass die vom Menschen bearbeiteten Aufgaben ausschliesslich kreativer Natur sein sollten, vielmehr sollte der Routineanteil durch den Einsatz von EDV auf ein sinnvolles Ausmass reduziert werden.

Tabelle 2: Vergleich von Mensch und Computer [LZG=Langzeitgedächtnis, KZG= Kurzzeitgedächtnis].

	Mensch	**Computer**
Stärken	sehr mächtige Fähigkeit zur Mustererkennung; Fähigkeit zur selektiven Aufmerksamkeitssteuerung; lebenslange Lernfähigkeit; 'unbegrenzte' Kapazität des LZG; assoziativer Zugriff auf Gedächtnisinhalte; komplexe multi-modale Speicherung; hohe Kontextsensitivität (z.B. bei Entscheidungen usw.); kreative Tätigkeiten;	permanente Verfügbarkeit des Speichers; zuverlässiger Speicherzugriff; relativ grosse Speicherkapazität; sehr schnelle Verarbeitungsgeschwindigkeit; 'fehlerfreie' Datenverarbeitung; Routineoperationen;
Schwächen	geringe Speicherkapazität des KZG; schnelle Vergessensrate des KZG; langsame Verarbeitungsgeschwindigkeit; fehleranfällige Verarbeitung; unzuverlässiger Zugriff auf das LZG.	sehr eingeschränkte Möglichkeit zur Mustererkennung; begrenzte Kapazität des 'LZG'; begrenzte 'Lernfähigkeit'; multi-modale Speicherung; geringe Kontextsensitivität.

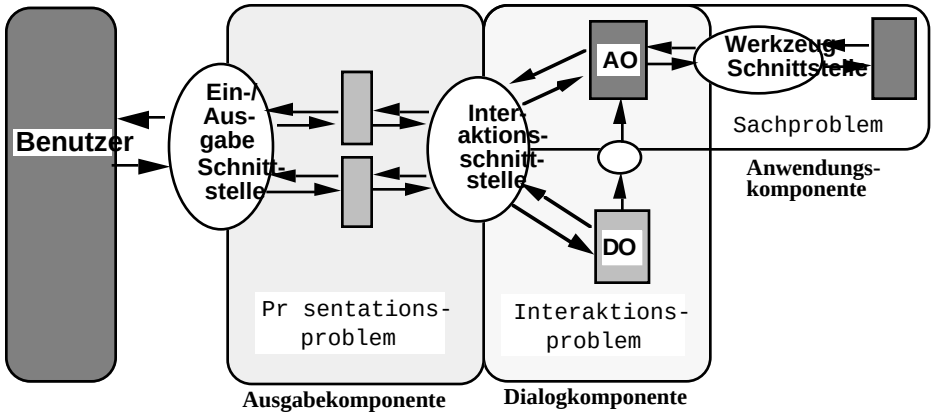


Abbildung 2: Übersicht über die verschiedenen Schnittstellen eines interaktiven EDV-Systems. [AO: Anwendungsoperatoren; DO: Dialogoperatoren]

Wenn zur Aufgabenbearbeitung interaktive EDV-Systeme eingesetzt werden, gibt es drei Problem- bzw. Gestaltungsbereiche: Das *Sachproblem*, das *Interaktionsproblem* und das *Präsentationsproblem* (siehe Abbildung 2).

Das *Sachproblem* besteht darin, dem Benutzer zur Unterstützung seiner Aufgabenbearbeitung alle notwendigen Funktionen und entscheidungsrelevanten Informationen auf einem geeigneten 'Granulations- und Darstellungsgrad' anzubieten (die Gestaltung der 'Werkzeugschnittstelle'). Diese aufgabenbezogenen Funktionen werden vom Benutzer durch die Eingabe entsprechender Anwendungsoperatoren (AO) ausgelöst bzw. aktiviert. Die Eingabe eines Dialog- bzw. Anwendungsoperators wird auch Dialog- bzw. Anwendungs-Operation genannt. Die vom System angebotenen Operationen und Informationen dürfen aber den Benutzer in seinem individuellen Problembearbeitungsprozess nicht unnötig einschränken. Die Lösung des Sachproblems bedeutet deshalb die aufgaben- und benutzerangepasste Gestaltung der Werkzeugschnittstelle mit allen notwendigen Anwendungsoperationen und -informationen zur Bearbeitung der gestellten Aufgaben.

Das *Interaktionsproblem* kommt im wesentlichen durch die Begrenzungen der Ausgabe- und Interaktionsschnittstelle zustande (siehe Streitz 1990). Damit der Benutzer (aufgabenbezogene) Informationen erhalten und die notwendigen Operationen durchführen kann, muss er die entsprechenden Anwendungs- bzw. Dialogoperatoren eingeben (AO bzw. DO). Hierzu stehen zur Zeit unterschiedliche Techniken zur Verfügung (Kommandos, Menüs, maussensitive Bereiche, PF-Tasten usw.; siehe Helander 1991).

Fast jede heutzutage existierende Benutzungsoberfläche verlangt nun vom Benutzer einige Dialogoperationen, welche lediglich zur Gestaltung der Ein-/Ausgabe- und der Interaktionsschnittstelle (z.B. Steuerung der Interaktion) dienen und nichts zur unmittelbaren Aufgabenbearbeitung beitragen. Der Unterschied zwischen diesen speziellen Dialogoperationen und den Anwendungsoperationen kann man sich am besten wie folgt merken: Alle Operationen, welche den Zustand des jeweiligen Anwendungsobjektes (z.B. Textdokument, Datenbank usw.) verändern, gehören zu den Anwendungsoperationen. Alle anderen Operationen (z.B. die Veränderungen von Fenstern, wie Öffnen, Schliessen, Verschieben usw.) sind 'reine' Dialogoperationen. Die Handhabung dieser Dialogoperationen sollte so einfach wie möglich sein und dem Benutzer keinen unnötigen zusätzlichen kognitiven Aufwand abfordern. (Dies darf jedoch nicht mit der Komplexität der Werkzeugschnittstelle und seiner Anwendungsoperationen verwechselt

werden, welche sich ausschliesslich nach der Komplexität der zu bearbeitenden Aufgaben zu richten hat!)

Das *Präsentationsproblem* besteht darin, die Ausgaben des Computers so zu gestalten, dass der Benutzer bei der Aufgabenbearbeitung mit allen notwendigen Informationen in seinem aktuellen Aufmerksamkeitsfokus auf der Ein-/Ausgabeschnittstelle versorgt wird und hierzu möglichst wenig Dialogoperationen benötigt (die Gestaltung der 'Ein-/Ausgabeschnittstelle'; siehe Abbildung 2). Die Ausgabe von aufgabenbezogenen Informationen auf der Ausgabeschnittstelle ist für den Benutzer das 'externe' Gedächtnis. Wie sich die drei Problembereiche gestalten lassen und was dies im einzelnen bedeuten kann, hängt von den spezifischen Eigenschaften menschlicher Wahrnehmungsleistungen, von Gedächtnisleistungen und den Handlungsmöglichkeiten ab.

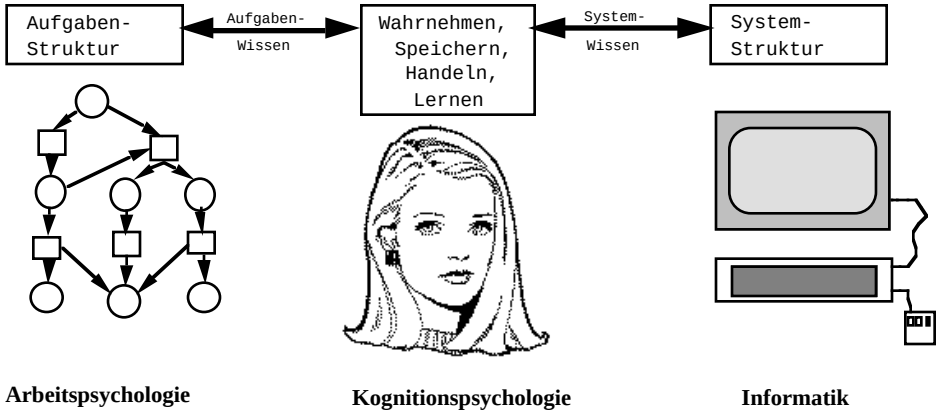


Abbildung 3: Übersicht über die drei verschiedenen Disziplinen.

Bei der benutzungsorientierten Systemgestaltung müssen Ergebnisse der Arbeitspsychologie, der Kognitionspsychologie und der Informatik integriert werden. Einerseits muss der Benutzer Wissen über die Bearbeitung von Aufgaben haben, andererseits muss er Wissen über die Handhabung des interaktiven Systems besitzen (siehe Abbildung 3).

Zunächst einmal sind die softwaretechnischen Kriterien der *Effizienz* und der *Zuverlässigkeit* von grundlegender Bedeutung. Die Entwicklung preiswerter Hardware und ihr Einsatz im Bereich der Arbeitsplatzrechner (PCs

und Workstation) lassen eine positive Entwicklung erkennen, bei der dem Benutzer eine möglichst grosse technische 'Power' zur Verfügung gestellt wird.

Tabelle 3: Übersicht über verschiedene Bewertungsmodelle der Mensch-Computer-Interaktion (siehe auch Reiterer 1990 und Ilg 1993).

Dzida, et al. (1977)	**DIN 66234 Teil 8 (1988)**	**VDI-Richtlinie 5005 (1990)**	**EG-Richtlinie 90/270/EWG (1990)**	**ISO 9241 Entwurf (1991)**	**Ulich (1994)**
	Aufgabenangemessenheit	Aufgabenangemessenheit	Tätigkeitsangepasstheit	Aufgabenangemessenheit	• Ganzheitlichkeit • Anforderungsvielfalt, • Autonomie, • soziale Interaktionsmöglichkeiten
Selbsterklärungsfähigkeit	Selbsterklärungsfähigkeit		Angaben über die jeweiligen Systemabläufe	Selbsterklärungsfähigkeit	• Transparenz, • Unterstützung
Rückkopplungsfähigkeit					• Feedback
	Erwartungskonformität		Format- und tempogerechte Informationsdarstellung	Erwartungskonformität	• Konsistenz • Kompatibilität
Dialogflexibilität		Handlungsflexibilität			• Flexibilität
einfache Anwendbarkeit				Erlernbarkeit	• Lernmöglichkeiten
		Kompetenzförderlichkeit	Unterrichtung und Unterweisung		• Entwicklungsmöglichkeiten
			Anpassbarkeit an Kenntnis- und Erfahrungsstand	Individualisierbarkeit	• individuelle Auswahl, • individuelle Anpassung
			Anhörung und Beteiligung		• Partizipation
	Steuerbarkeit			Steuerbarkeit	• Flexibilität
Zuverlässigkeit	Fehlerrobustheit			Fehlertoleranz	[wird vorausgesetzt]

Erst wenn diese grundlegenden Eigenschaften des technischen Systems gewährleistet sind, lassen sich sinnvolle Gestaltungsmassnahmen aus arbeitspsychologischer Sicht für die interaktive Software umsetzen. Es wird daher im weiteren vorausgesetzt, dass das technische System *keine wesentlichen Beschränkungen* hinsichtlich Antwortzeit und Speicherplatz aufweist und auf jeden Fall *zuverlässig* arbeitet.

Arbeitspsychologische Kriterien lassen sich den verschiedenen Phasen eines vollständigen Handlungsvollzuges zuordnen. Für die Unterstützung der *Orientierung* ([1] Zielsetzung, [2] Planung, [4] Kontrolle) zu Beginn und während einer Handlung sind die Kriterien *Transparenz*, *Feedback*, *Kompatibilität* und *Konsistenz* sowie der Bereich *Unterstützung* wichtig.

Die verschiedenen Bewertungsmodelle der Mensch-Computer-Interaktion (Dzida et al. 1977; DIN 66234 1988; VDI 5005 1990; ISO 9241 1991; Ulich 1994) zeigen unterschiedliche Schwerpunkte und Differenzierungsgrade (siehe Tabelle 3). Das Modell von Ulich (1994) ist nicht nur handlungspsychologisch begründbar und empirisch abgesichert, sondern auch am weitesten differenziert. Die meisten Übereinstimmungen mit dem Modell von Ulich finden sich in der EG-Richtlinie und der ISO 9241.

Kasten 1: Ein Beispiel für das Kriterium *Transparenz*.

> Benutzer können erkennen, ob ein eingegebener Befehl behandelt wird oder ob das System auf weitere Eingaben wartet. Bei längeren Vorgängen sollte das System Zwischenzustandsmeldungen abgeben [können].

Die *Transparenz* eines Systems soll dem Benutzer die Bildung eines Struktur- und Prozessmodells des Systems im Gedächtnis erleichtern, was ihm die notwendigen Orientierungsgrundlagen für die Benutzung bietet. Der Benutzer muss sich ja über den Inhalt (Funktionen bzw. Operationen, Informationen) und dessen Organisation im System (Zugriffspfade, Ordnungskriterien) generell sowie über den aktuellen Dialogzustand ein 'Bild' machen können, um die technische Unterstützung auch effizient nutzen zu können. Ein System sollte dementsprechend seine Nutzungs- und Kontrollmöglichkeiten sowie aktuelle Bearbeitungszustände dem Benutzer an der

Benutzungsoberfläche strukturiert und verständlich offenlegen (siehe auch Maass 1994).

Nach Wandke (1988, S. 4) kann es für den Benutzer – insbesondere bei abfragenden Systemen mit 'prompting-technique' – in Ermangelung erkennbarer Ziele und eines Bezugssystems zu Ziel- und Orientierungslosigkeit sowie einer Überlastung des Kurzzeitgedächtnisses kommen, da "alle vorausgegangenen und nachfolgenden Nutzereingaben und Systemausgaben, die einen Einfluss auf die aktuelle Nutzereingabe haben, im Arbeitsgedächtnis behalten werden müssen". Typische Benutzeraussage: "Wenn ich etwas zurückverfolgen will, muss ich immer wieder von Bild zu Bild gehen, um die Informationen abzurufen, vorher habe ich einfach meine Papiere durchgeblättert und hatte alles auf *einen Blick.*" Insgesamt sind integrale, parallele Informationsdarstellungen gegenüber sequentiell dargebotenen Informationen von Vorteil. Benz und Haubner (1983) demonstrierten eindrücklich den positiven Einfluss, den nach Gestaltgesetzen strukturierte, 'geordnete' Bildschirm-Masken auf Leistung, Beanspruchung und Akzeptanz der Benutzer ausüben. Der positive Effekt strukturierter Menüs ist bekannt. Die höhere Transparenz der direktmanipulativen Benutzungsoberfläche gegenüber einer menü- und funktionstastenorientierten Oberfläche ist u.a. auf den grösseren, permanent sichtbaren Informationsgehalt sowie eine deutlich grössere Flexibilität in der Interaktion zurückzuführen.

Zusammengefasst ist festzuhalten, dass die *Transparenz* ein zentrales, aber schwer fassbares bzw. isolierbares Kriterium darstellt, das viele Bezüge zu anderen Kriterien – wie Konsistenz, Unterstützung, Feedback – aufweist und ausserdem stark von der Erfahrung des Benutzers abhängt.

Kasten 2: Ein Beispiel für das Kriterium *Feedback*.

> Die für die Planung und Ausführung von Tätigkeiten zentralen Rückmeldungen (Feedback) werden im aktuellen Dialogkontext für den Benutzer wahrnehmbar ausgegeben.

Zu den wesentlichen Merkmalen des Kriteriums *Feedback* im Rahmen einer vollständigen Aufgabenbearbeitung gehören: 'Ausführungsfunktionen mit Ablauffeedback zur allfälligen Handlungskorrektur' sowie 'Kontrolle mit Resultatfeedback und der Möglichkeit, Ergebnisse der eigenen Handlungen auf Übereinstimmung mit den gesetzten Zielen zu überprüfen'.

Im Bereich der Softwareergonomie haben sich verschiedene Untersuchungen mit einer speziellen Form von Rückmeldung – den Fehlermeldungen – beschäftigt, die erst dann erforderlich wird, wenn der Benutzer eine Aktion nicht systemgerecht durchgeführt hat. So hat Norman (1983) eine differenzierte Fehlerklassifikation vorgestellt und Hinweise für die Gestaltung von Schnittstellen und Fehlermeldungen abgeleitet. Dabei hat sich gezeigt, dass geeignetes Feedback zur Vermeidung mancher Fehler beitragen würde. Fehler sollten allerdings nicht um jeden Preis vermieden bzw. vom System toleriert werden, sondern auch gezielt für Lernprozesse eingesetzt werden. Zapf und Frese (1989) betrachten Fehler als 'mismatch' (Unangepasstheit) von Aufgabe und Computer (Funktions- bzw. Sachproblem) sowie von Computer und Benutzer (Nutzungs- bzw. Interaktionsproblem). Die meisten Fehler treten bei Operationen auf, die ein hohes Mass an Vorausplanung erfordern, wenn kein geeignetes Feedback gegeben wird. Fehlermeldungen mit mangelnder Angabe von Fehlerursache und Korrekturmöglichkeiten werden von den Benutzern auch dementsprechend schlecht beurteilt.

Kasten 3: Beispiele für das Kriterium *Konsistenz*.

> Das System sowie dessen *Antwortverhalten* sollten für Benutzer transparent und konsistent sein. Ähnliche Aktionen sollten ähnliche Ausführungen bewirken, andernfalls muss dies durchschaubar gemacht werden.
> Z.B. die Belegung der Funktionstasten, Menüoptionen, Schaltflächen auf dem Bildschirm sollte über alle Dialogkontexte hinweg innerhalb eines Programmes wie auch über die anderen verwendeten Programme hinweg einheitlich und damit konsistent sein.

Das Kriterium der *Konsistenz* bezieht sich auf die Berechenbarkeit des Systemverhaltens, wodurch Erwartungen des Benutzers erfüllt und Überraschungseffekte vermieden werden. Ein konsistent-regelhafter Aufbau von Dialogstruktur, Semantik und Syntax der Operationen (z.B. Kommandos) ermöglicht dem Benutzer die Rekonstruierbarkeit und entlastet dadurch das Gedächtnis von faktisch präziser und damit aufwendiger Speicherung. Generell lässt sich sagen, dass ein System um so konsistenter ist, je weniger Regeln man zur Beschreibung des Systems benötigt. Norman (1983) stellte fest, dass Inkonsistenzen in der Systemhandhabung die Quelle mancher Benutzerfehler sind.

Unter dem Kriterium *Kompatibilität* versteht man den Grad an Übereinstimmung zwischen den semantischen Gedächtnisstrukturen des Benutzers und den – durch die Aufgabenerfordernisse bestimmten – Benutzeraktionen. Norman (1989) spricht in diesem Zusammenhang auch von *natürlichen Mappings*, womit die Nutzung äusserer Analogien, Metaphern und kultureller Standards gemeint ist, welche zu einem *unmittelbaren Verständnis* führen (z.B. Drehen des Autolenkrades im Uhrzeigersinn lenkt das Auto nach rechts usw.). Da viele natürliche Mappings *analoger Natur* sind, empfiehlt sich oftmals der Einsatz von grafischen, *direkt-manipulierbaren Benutzungsoberflächen*.

Kasten 4: Ein Beispiel für das Kriterium *Kompatibilität*.

> Bei der *Darstellungsform* für Einzelinformationen sollte ebenso wie für ganze Bilder ggf. auf Übereinstimmung mit entsprechenden gedruckten Vorlagen oder Unterlagen geachtet werden. *Sprache* und begriffliche Komplexität des Dialoges sollten an den Gepflogenheiten und Kenntnissen des spezifischen Benutzerkreises orientiert sein. Anstelle von EDV-Kürzeln sollte mit den jeweils fachspezifischen Begriffen der Benutzer gearbeitet werden können.

Eine direkte Manipulation erfolgt durch analoges Bewegen, Verändern oder in eine bestimmte Relation Setzen (z.B. Anwendung einer Funktion) von grafisch visualisierten Objekten auf dem Bildschirm. Die analogen Operationen werden in der Regel durch ein entsprechendes Zeigegerät (z.B. Maus, Joystick usw.) ausgeführt. Da der Benutzer Objekte in einer Scheinwelt (z.B. die 'Desktop'-Metapher) direkt manipulieren kann, hat er oftmals den Eindruck, mit realen Objekten zu agieren. Direktmanipulierbare Benutzungsoberflächen zeichnen sich durch folgende Eigenschaften aus:

(1) Die permanente Sichtbarkeit der relevanten Objekte;

(2) schnelle, umkehrbare, einstufige Operationen mit unmittelbarer Rückmeldung und

(3) die Ersetzung komplexer, 'digitaler' Kommandos durch physische, 'analoge' Aktionen (z.B. Mausbewegungen, Selektionsoperationen usw.).

Zuerst wählt der Benutzer das relevante Objekt (O) aus und wendet dann eine entsprechende Funktion (F) darauf an. Diese Art der Interaktion nennt man daher auch O-F-Struktur. Im Gegensatz dazu herrscht bei den traditionellen Benutzungsoberflächen (Frage-Antwort, Kommando, Menü, Formular-Dialog) meistens die F-O-Struktur vor. Sehr viele alltägliche Handlungen mit Objekten der realen Welt sind jedoch nur mit der O-F-Struktur kompatibel.

Um eine entsprechende Kompatibilität auf der Ein-/Ausgabeschnittstelle zu erreichen, werden zur Zeit multimediale Benutzungsoberflächen eingesetzt. Multimediale Benutzungsoberflächen bedingen die Ausarbeitung entsprechender Metaphern und die dramaturgische Erarbeitung nichtsequentieller Vermittlungsformen. Multimediale Benutzungsoberflächen erhöhen drastisch die Chancen, dass sich die aufgabenangemessene Darstellung der Bearbeitungsobjekte auf eine benutzungsgerechte Systemnutzung positiv auswirken kann. Dabei führt das multimediale Design zur Auflösung und Neufassung der traditionellen Gestaltungsregeln der verschiedenen, beteiligten Einzelmedien: Text, Bild, Ton und Film. Beim Informationstransfer zwischen den Medien werden nicht nur Daten transportiert, viel wesentlicher ist die expressive und inhaltliche Gestaltung, die sich durch die Übertragung von Information in ein neues Darstellungsmedium ergibt. So ändert sich z.B. die Wirkung numerischer Daten, sobald diese in eine grafische Darstellung übertragen werden, mit der sich sogar interaktiv bestimmte Simulationen durchführen lassen. In dieser Wirkungsverschiebung liegen die herausragenden Vorteile multimedialer Benutzungsoberflächen.

Durch die Kompatibilität soll eine Reduktion mentaler Transformationsschritte des Benutzers und damit eine Entlastung des mit beschränkter Kapazität arbeitenden Kurzzeitgedächtnisses erreicht werden. Ulich (1986) hat in diesem Zusammenhang eine sinnvolle Unterscheidung in *Sprach*- und *Darstellungskompatibilität* vorgenommen. So haben Untersuchungen gezeigt, dass die Gestaltung von Dialogsprachen in Anlehnung an die natürliche (Fach-)Sprache (Vokabular; Abkürzungen; Reihenfolge von Subjekt, Verb, Objekt usw.) des Benutzers von Vorteil sein kann.

Das Kriterium *Unterstützung* bezieht sich auf rechnergestützte Lern- und Arbeitshilfen, die dem Benutzer über die normalen Dialogmeldungen hinausgehende Informationen zur Bewältigung von Problemsituationen zur Verfügung stellen. Diese Lern- und Arbeitshilfen sollen dem Benutzer die Orientierung erleichtern und ihm helfen, Software besser zu durchschauen (Informationen über wichtige Zusammenhänge, Erklärung von Ereignis-

sen, Handlungsergebnissen usw.) sowie Dialogabläufe besser vorherzusehen (Bilden von Hypothesen und Prognosen) und gemäss eigenen Zielen zu beeinflussen. *Unterstützung* muss aufgabenorientiert gestaltet sein und sich auf die Dialog- und Anwendungskomponente der Benutzungsoberfläche beziehen. Die Kontrolle über die rechnergestützten Lern- und Arbeitshilfen (z.B. Bestimmung des Umfangs der Hilfemeldungen) soll beim Benutzer und nicht beim System (Adaptivität) liegen. Möglichkeiten der Unterstützung von Benutzern werden unter so unterschiedlichen Begriffen wie Selbsterklärungsfähigkeit, 'advice-giving systems', 'on-line assistance', 'users support' oder 'user assistance' beschrieben. Sie lassen sich folgenden drei Grundformen zuordnen: Computerunterstützte Handbücher, Hilfesysteme und Tutorials.

Kasten 5: Ein Beispiel für das Kriterium *Unterstützung*.

> *Dialoghilfen* sowohl zu inhaltsbezogenen wie zu vorgehensbezogenen Aspekten sollten von den Benutzern während des Dialogs jederzeit abgerufen werden können. Das Betätigen einer allfälligen Help-Taste sollte gegenüber anderen Befehlen einen Sonderstatus einnehmen. Das System sollte eine *Rückfragemöglichkeit* derart bereitstellen, dass auf eine Aufforderung durch die Benutzer hin ggf. ausführlichere, aufgabenbezogene Antworten abgegeben werden.

Das elektronische, am Bildschirm zugängliche Handbuch wird verschiedentlich als Alternative zum traditionellen Handbuch angesehen. Der Hauptnachteil der am Bildschirm zur Verfügung gestellten Information liegt in der im Vergleich zur gedruckten Version deutlich schlechteren Lesbarkeit. Grössere Textpassagen sind deshalb immer in einem gedruckten Benutzerhandbuch zur Verfügung zu stellen. Hinweise zur Gestaltung gedruckter Handbücher und Bedienungsanleitungen befinden sich z.B. in Rupietta (1987) und Boedicker (1990).

Rechnergestützte Auskunfts- und Informationssysteme, die dem Benutzer Informationen über Kommandos, Abfragen oder Dialogzustände zur Verfügung stellen, sollten immer verfügbar, richtig und vollständig sein und ein – hinsichtlich Leseniveau, Umfang und Länge – einheitliches Format haben. Wichtig ist in diesem Zusammenhang, dass die Inhalte möglichst kontext-sensitiv, das heisst, auf den konkreten Bearbeitungszustand ausge-

richtet sind. An den meisten Hilfesystemen wird bemängelt, sie seien für Programmierer geschrieben und deshalb zu abstrakt, zu algebraisch und voll von Computerjargon. So lässt sich u.a. die Sprache dem Benutzer anpassen – Kriterium der Sprachkompatibilität – und statt systemorientierter Meldungen können aufgabenorientierte Informationen zur Verfügung gestellt werden. Die Ergebnisse empirischer Studien zeigen deutlich, dass auch Computerlaien von On-line-Hilfesystemen profitieren können, wenn auch die Hilfemeldungen den Kriterien benutzungsorientierter Dialoggestaltung entsprechen.

Die nun folgenden Kriterien *Flexibilität, individuelle Auswahl* und *individuelle Anpassung* beziehen sich primär auf die Planung, Auswahl der Mittel und Ausführung einer Handlung (siehe Teil A, Kapitel 1.2).

Kasten 6: Das Kriterium *Flexibilität*.

> Flexibilität eines interaktiven Systems wird durch das vorhandene Ausmass an Handlungsspielraum festgelegt. Der Handlungsspielraum ist die Summe (objektiv) vorhandener Freiheitsgrade zur selbständigen Setzung und Erreichung von (Teil-)Zielen durch variable Abfolge von (Teil-)Schritten.
> Der Benutzer kann aus einer vorgegebenen Menge von Alternativen auswählen und beliebig zwischen den verschiedenen Alternativen wechseln, da ihm alle Möglichkeiten jederzeit zur Auswahl zur Verfügung stehen (z.B. Tastatur und Maus).

Flexibilität gilt wohl zu Recht als eines der zentralen Kriterien der Dialoggestaltung, entscheidet sich doch an der Flexibilität bzw. Rigidität eines Systems, ob sich der Mensch als Benutzer oder als Bediener des Computers fühlt. Mit der Flexibilität ist die Beeinflussbarkeit des Systemverhaltens im Sinne objektiv vorhandener Freiheitsgrade – vermittelt über den Handlungsspielraum – gemeint. Der Handlungsspielraum dient zur selbständigen Setzung und Erreichung von (Teil-)Zielen durch variable Abfolge von (Teil-)Schritten. Die Steuerung des Dialogablaufes – und damit die Reihenfolge von Operationen und Arbeitsschritten – sollte in inhaltlicher und zeitlicher Hinsicht sowie bezüglich einzusetzender Mittel weitgehend dem Benutzer überlassen werden. Dadurch können unterschiedliche Vorgehensweisen – die durch sich ändernde Aufgabenerfordernisse und/oder unter-

schiedliche Arbeitsweisen der Benutzer bedingt sein können – ermöglicht werden.

Der Bedeutung des Kriteriums entsprechend haben sich auch weltweit Wissenschaftler damit befasst und sind zu mehrheitlich übereinstimmenden Ergebnissen gekommen: Flexibilität eines Dialogsystems ist unbedingt notwendig, damit der Benutzer in weiten Grenzen Teilaufgaben selbständig definieren, deren Bearbeitungsreihenfolge festlegen und auch bei unvorhergesehenen Ereignissen seine Pläne ändern kann. Der Benutzer sollte den Dialogablauf nach seinen kognitiven Prozessen ausrichten können, selbst assoziative Gedankensprünge sollten durch Sprungmöglichkeiten im Dialog realisierbar sein. Demgegenüber erzwingen rigide Dialogstrukturen eine Anpassung der kognitiven Prozesse des Benutzers an die im System festgelegte, abstrakt-logische Aufgabenstruktur, was sich in schlechteren Leistungen und vor allem auch im subjektiven Erleben der Benutzer in Form von Unzufriedenheit und geringer Akzeptanz des Systems ausdrückt. Benutzer derartiger Systeme fühlen sich laut ihren Aussagen in ein Schema gepresst, das ihren Bedürfnissen und ihrer Vorgehensweise bei der Aufgabenbewältigung nicht gerecht wird und sie zum Bediener des Computers degradiert.

Präzisierend ist allerdings zu ergänzen, dass Anfänger eine stärkere Systemführung – und damit geringere Flexibilität – begrüssen, nach relativ kurzer Zeit aber schon das Bedürfnis nach grösseren Kontrollmöglichkeiten äussern, wie dies in einigen empirischen Studien zum Ausdruck kommt. Dialogflexibilität ist offensichtlich nicht nur notwendig, um unterschiedlichen Vorgehensweisen der Benutzer Rechnung zu tragen, sondern darüber hinaus werden dadurch auch individuelle Bearbeitungsweisen berücksichtigt, die sich durch Lernprozesse über die Zeit ergeben.

Kasten 7: Das Kriterium *individuelle Auswahlmöglichkeit*.

> Der Benutzer kann das Systemverhalten durch die Einstellung von Systemparametern (Schaltern usw.) auf seine individuellen und aufgabenbezogenen Bedürfnisse abstimmen. Der Benutzer kann aus einer vorgegebenen Menge von Alternativen eine der angebotenen Möglichkeiten auswählen; anschliessend ist jeweils nur diese ausgewählte Möglichkeit aktiv und benutzbar.

Im Unterschied zur Flexibilität eines Systems, die für Gruppen von Benutzern Freiheitsgrade für unterschiedliche Vorgehensweisen ermöglicht, soll die *individuelle Auswahlmöglichkeit* und die *individuelle Anpassung* eines Systems allen Benutzern die Möglichkeit bieten, die Benutzungsoberfläche nach ihren persönlichen Bedürfnissen massgeschneidert einzurichten; im Englischen werden dafür häufig die Begriffe 'taylorized', 'customized', 'personalized' bzw. 'individualized' gebraucht.

Es sollte dem Benutzer eines Systems möglich sein, durch einfache Veränderung einiger Parameter bzw. Schalter für Systemeinstellungen die seinen Kenntnissen und Bedürfnissen gerechte Formen der Interaktion selbst zu realisieren (z.B. Darstellung von Informationen, Belegung von Funktionstasten usw.). In der Praxis haben sich Systeme bewährt, deren Benutzungsoberfläche unterschiedliche Interaktionsformen – z.B. Menü und Kommando, Maus und Tastatur – umfassen, die je nach Erfahrung des Benutzers von ihm selbst ausgewählt werden können.

Im Unterschied zur Flexibilität bietet die *individuelle Auswahl* alternative Nutzungsvarianten an, zwischen denen sich der Benutzer im Sinne von Wahlmöglichkeiten entscheiden kann. Sobald er jedoch seine Entscheidung getroffen hat, ist sein weiterer Bearbeitungsweg festgelegt und kann (im Gegensatz zur Flexibilität) nicht mehr unmittelbar neu gewählt werden. Erst nach einem zusätzlichen Auswahldialog (z.B. Setzen verschiedener Schalter der Systemeinstellungen) kann eine andere Nutzungsvariante aktiv werden.

Kasten 8: Das Kriterium *individuelle Anpassung*.

> Individuelle Anpassungsmöglichkeiten bieten die Möglichkeit der eigenständigen Gestaltung ('Eigenprogrammierung') und dementsprechend auch der Erweiterung objektiver Handlungsspielräume. Der Benutzer kann die Menge von Benutzungsalternativen erweitern.

Eine weitere Möglichkeit der Individualisierbarkeit wird als *individuelle Anpassung* bezeichnet. In diesem Fall erlaubt das System dem Benutzer Änderungen am System selber vorzunehmen (Adaptierbarkeit), oder das System verändert sich selbst aufgrund der Nutzungsart des Benutzers (Autoadaptierbarkeit, siehe Haaks 1992). Bei autoadaptiven Systemen sollte der Benutzer Kontrolle über diese Autoadaptierbarkeit haben.

Es zeigte sich im Rahmen einer empirischen Studie, dass bei der Benutzung eines simulierten Informationsabfragesystems eine fast doppelt so hohe Leistung von Benutzern erreicht wurde, wenn sie das jeweils aktive Deskriptorensystem individuell selbst entwickelt hatten. Demgegenüber musste eine Kontrollgruppe mit einer vorgegebenen 'Fremdkonstruktion' des Deskriptorensystems arbeiten.

Ein Dialogsystem ist dann benutzungsfreundlich, wenn dem Benutzer alle zur Aufgabenerfüllung benötigten Informationen und Funktionen in einer der verlangten Arbeitshandlung entsprechenden Form und Reihenfolge angeboten werden. Es konnte gezeigt werden, dass dem Benutzer für manipulative Operationen – wie z.B. Editieren von Texten – besser bildliche Symbole dargeboten werden sollten, für die Auslösung sequentiell strukturierter Ereignisse – wie z.B. Dateihandhabung – dagegen Kommandoworte angemessener sind. Ebenso fand sich ein Zusammenhang zwischen dem Aufgabentyp und einer Dialogtechnik, die dem Benutzer effizientes Arbeiten erlaubte. Die Frage der Gestaltung von Fenstern – überlappende versus nicht überlappende Fenster – ist nur in Abhängigkeit von der zu unterstützenden Aufgabe zu beantworten. Die Abhängigkeit der Dialogstruktur vom Aufgabengebiet und vom Arbeitsablauf des Benutzers wurde insbesondere in Untersuchungen zur Strukturierung von Menüsystemen nachgewiesen. Es konnte schliesslich noch gezeigt werden, dass mit steigendem Wiederholungsgrad und zunehmender Komplexität einer Aufgabe die Benutzer vermehrt von den Möglichkeiten zur Bildung von Makros Gebrauch machen.

1.4 FAZIT

Erst wenn die Benutzungsoberfläche insgesamt software-ergonomisch benutzungsgerecht gestaltet ist und dadurch das Interaktionsproblem minimiert wurde, ist die individuelle Nutzungsweise kein erschwerendes Hindernis, sondern eine echte Unterstützung und Entlastung des Benutzers bei seiner Aufgabenerfüllung. Zusammengefasst kann man sagen, dass Normen, Kriterien und Richtlinien einerseits notwendige Hinweise, wie benutzungsgerechte Oberflächen zu gestalten sind, liefern, andererseits aber auch die Möglichkeit bieten, Schnittstellen zu vereinheitlichen. Richtlinien erfüllen folgende zentrale Funktionen:

(1) Sie sind Hilfsmittel, individuelle Beiträge zur Schnittstellengestaltung zu koordinieren und zu vereinheitlichen.

(2) Entscheidungen werden nur einmal getroffen und nicht immer wieder von einzelnen Beteiligten verändert.

(3) Anhand von Richtlinien lassen sich detaillierte Anforderungskataloge erarbeiten.

(4) Die erstellten Anforderungskataloge können gleichzeitig als Grundlage für die Evaluation der Zwischenstadien und Endprodukte dienen.

(5) Richtlinien können als Grundlage für die Koordination der Zusammenarbeit verschiedener am Projekt beteiligter Gruppen dienen.

Im Rahmen dieses Kapitels wurden arbeitspsychologische Grundlagen und begründete Kriterien für die Gestaltung von Arbeitssystemen, Arbeitsaufgaben und Benutzungsoberflächen dargestellt. Diese Kriterien sind eine wichtige Orientierungshilfe für die Gestaltung technisch-organisatorischer Strukturen und Abläufe, die den Ansprüchen humaner und wirtschaftlich effizienter Arbeit genügen sollen. Diese Kriterien ersetzen allerdings keine Beteiligung von Mitarbeitern bei technisch-organisatorischen Veränderungen. Diese Kriterien können vielmehr einen solchen Beteiligungsprozess unterstützen, indem die Gestaltungshinweise dieser Kriterien mit den Beschäftigten für den konkreten Arbeits- und Aufgabenkontext spezifiziert und umgesetzt werden. Vor diesem Hintergrund werden im nächsten Kapitel Grundlagen der Benutzerbeteiligung vermittelt.

2 Grundlagen der Benutzerbeteiligung

In diesem Kapitel werden zuerst die Gründe für eine Beteiligung von Benutzern bei der Softwareentwicklung und die Voraussetzungen dazu beschrieben. Anschliessend werden spezifischer die unterschiedlichen Möglichkeiten der Kooperation von Benutzern und Softwareentwicklern dargestellt (siehe dazu auch Mambrey und Oppermann 1983, Mumford und Welter 1984, Kissler 1988, Koslowski 1988, Mambrey, Oppermann und Tepper 1986 sowie Peschke 1986).

2.1 GRÜNDE FÜR BENUTZERBETEILIGUNG

Vielfältige Praxiserfahrungen lassen folgende mögliche Vorteile von Benutzerbeteiligung erwarten.

Für den Anwender:

(1) Bessere Aufgabenangemessenheit der Software,

(2) höhere Akzeptanz durch Benutzerorientierung,

(3) geringere Kosten durch Vermeidung nachträglicher, kostenintensiver Korrekturen.

Für die Benutzer:

(1) Einsicht in Verständnis für Möglichkeiten und Grenzen neuer Technologien und die Arbeit der Entwickler,

(2) grössere Identifikation mit bzw. höhere Akzeptanz der Lösung,

(3) höhere Motivation zur Benutzung des Systems.

für die Entwickler:

(1) Feedback über ihr Produkt,

(2) Qualifizierungsmöglichkeiten in Fachfragen,

(3) bessere Einsicht in die Arbeit der Benutzer,

(4) grössere Sicherheit bei der Lösungsfindung.

2.1.1 Widerstand gegen Veränderung – Akzeptanz und Identifikation

Bei technisch-organisatorischen Veränderungen in Unternehmen sind auf allen hierarchischen Ebenen immer wieder *Ängste und Widerstände* bei den von der Veränderung Betroffenen zu beobachten, die sich auf verschiedene Weise manifestieren können: Minderung der Arbeitsleistung, Vermehrung der Absenzen, Gesuche um Versetzung, Stellenwechsel usw. Was sind aber die *Ursachen von Widerstand gegen Veränderungen?*

> Die wahre Natur des Widerstands: Unsicherheit und Ungewissheit über die *sozialen* Veränderungen.

Widerstand bezieht sich zumeist nicht auf die technischen, sondern auf die *sozialen Veränderungen*, mit denen in der Folge von angekündigten technisch-organisatorischen Veränderungen gerechnet wird. Hierzu zählen unter anderem die Veränderungen bezüglich zwischenmenschlicher Beziehungen, Selbständigkeit, Arbeitsinhalt, Status usw. Gerade die Einführung neuer Technologien löst bei den Betroffenen häufig Unsicherheit und Ungewissheit aus und führt zu Spekulationen über die zukünftige Arbeitssituation. Der Mensch hat aber ein grundlegendes Bedürfnis nach Klarheit über Ereignisse und Prozesse, die in seiner Umwelt ablaufen und ihn betreffen; zudem möchte er auch korrigierend eingreifen können, falls etwas seinen Interessen oder Ansichten zuwiderläuft. Auf betriebliche Veränderungen bezogen heisst dies:

Die Mitarbeiter möchten

(1) die Gründe für eine Innovation verstehen,

(2) die Folgen für die Veränderung ihrer Arbeitsbedingungen abschätzen und auch

(3) an der Veränderung mitwirken können, um ihre Meinung, ihre Erfahrungen und ihr Wissen einzubringen.

Bei betrieblichen Innovationsvorhaben ist allerdings häufig zu beobachten, dass vorwiegend Experten bzw. Spezialisten eingesetzt werden, welche eine Neuerung vorbereiten und einführen. Ihr formeller Auftrag bezieht sich häufig ausschliesslich auf technische und/oder organisatorische Probleme, zu deren Bewältigung *keine formellen* Kommunikationskanäle zu

den Betroffenen vorgesehen sind. Dies kann dazu führen, dass die Experten sich einseitig nur mit den *technischen* Aspekten neuer Ideen beschäftigen, sich vorwiegend an ihren eigenen Interessen und Vorstellungen orientieren und das Fachwissen der Betroffenen, deren Kenntnisse und Fertigkeiten aus der täglichen praktischen Erfahrung, unterschätzen.

> *Möglichkeiten zur Verringerung von Widerstand:* Information, Ausbildung und Beteiligung der Betroffenen!

Angesicht dieser Überlegungen scheinen die erforderlichen Massnahmen bei der Planung technisch-organisatorischer Neuerungen naheliegend zu sein. Durch eine frühzeitige und fortlaufende *Information* der Betroffenen über die Innovation können Ungewissheiten reduziert sowie Spekulationen und Gerüchtebildungen weitgehend vermieden werden.

Über die Information hinaus kann Befürchtungen der Betroffenen, die zukünftigen Anforderungen mit den bisherigen Kenntnissen und Fertigkeiten nicht mehr bewältigen zu können, durch rechtzeitige *Ausbildungsmassnahmen* begegnet werden.

Information und Ausbildung befähigen die Betroffenen auch zur *Mitwirkung an der Veränderung*. Für die Organisation ergibt sich dadurch die Chance, die in der täglichen Praxis erworbenen Erfahrungen der Mitarbeiter bezüglich einer effizienten Erfüllung der Arbeitsaufgaben zur Neugestaltung der Arbeitsorganisation zu nutzen. Die Nutzung dieses Fachwissens kann einen wesentlichen Beitrag zu Optimierung von Arbeitsabläufen und organisationalen Strukturen leisten. Ausserdem ist die Identifikation der Beteiligten mit dem Endprodukt grösser, wenn sie ihre Vorschläge verwirklicht sehen, und damit auch die Motivation, mit dem System zu arbeiten; das System erfährt eine insgesamt höhere Akzeptanz.

An dieser Stelle sei aber auch ausdrücklich vor jeder Form von *Pseudo*partizipation gewarnt: Solange man Partizipation nur als ein Mittel betrachtet, jemanden dazu zu bewegen, das zu tun, was man von ihm erwartet, wird diese Art der Pseudopartizipation nie befriedigende Ergebnisse zeitigen. Wirkliche Partizipation beruht stets auf Respekt.

Dabei darf jedoch nicht vergessen werden, dass die konkrete Gestaltung operativer Massnahmen eine gewichtige Bedeutung für den Erfolg oder Misserfolg eines Projektes besitzt. So können z.B. die Wahl einer ungün-

stigen Abteilung für den Einführungsversuch, die mangelnde Berücksichtigung von Verflechtungen eines Teilsystems mit dem Gesamtsystem oder der ungenügende Einbezug der Vorgesetzten ein Veränderungsprojekt zum Scheitern verurteilen, obwohl eine Beteiligung der Mitarbeiter praktiziert wird.

2.1.2 Fachliche Komplexität der Anwendung – Fachkompetenz der Benutzer

Bisherige, konventionelle Anwendungen von EDV-Systemen befassten sich mit der (Teil-)Automatisierung von gut formalisierbaren Routineaufgaben, dem Massengeschäft, mit vorwiegend einheitlichen Bearbeitungsverfahren. Demgegenüber unterstützen heutige Anwendungen vor allem ein breites Spektrum unterschiedlichster, schwer formalisierbarer und gering strukturierter Tätigkeiten mit individueller Aufgabenabwicklung. Anstelle der Bearbeitung von Routinefällen tritt dabei die Bearbeitung von Klassen von Sachfällen bzw. Einzelfällen in den Vordergrund. Zusätzlich erschwerend wirkt die hohe Veränderungsdynamik qualitativer, das heisst, schlecht formalisierbarer Anforderungen. Es wäre deshalb unrealistisch und verfehlt, von einem Softwareentwickler zu erwarten, dass er diese Komplexität ohne Mitarbeit der Benutzer bewältigen kann. Die Vielfalt moderner Entwicklungstools und technischer Möglichkeiten im allgemeinen erfordern von ihm in seinem Fachgebiet bereits hohe Qualifikationen!

Kasten 9: Beispiel für die Notwendigkeit von Benutzerbeteiligung.

> Eine Regel besagt, dass Optionen in einem Menü nach einem Kriterium (Inhalt, Sachlogik, Arbeitsablauf, Benutzungshäufigkeit, Alphabet) gruppiert sein müssen. Welches Kriterium im konkreten Fall Anwendung findet, kann am besten mit den Benutzern geklärt werden!

Damit ist ein Hauptgrund für die Beteiligung von Benutzern angesprochen: Das fachliche Erfahrungswissen der Benutzer kann durch nichts ersetzt werden! Insbesondere bei der Komplexität der heutigen Anwendungen verhelfen die Beiträge der Benutzer zur Entwicklung eines Systems, das bezüglich Funktionalität und Informationsbestand den Aufgaben der Benutzer und deren Arbeitsstil besser angepasst ist, als wenn es ein Entwickler 'im

stillen Kämmerlein' entwirft. Ausserdem erhält der Entwickler ein grösseres Sicherheitsgefühl. Er muss sich bei Problemen, offenen Fragen, Zweifelsfällen usw. nicht mehr den Kopf zerbrechen, 'wie das ein Benutzer sehen würde', sondern kann die Situation gemeinsam mit den Benutzern klären.

Sicherlich gibt es schon viele Fachbücher, die mit Kriterien, Richtlinien und Regeln dem Softwareentwickler Hilfe bei der Systementwicklung versprechen. Sie bieten ihm in der Regel einen Rahmen, eine Orientierungshilfe und allgemeine Hinweise zur *formalen Gestaltung* der Benutzungsoberfläche. Aufgrund ihres allgemeingültigen Anspruchs müssen sie aber notgedrungen die *inhaltliche Seite* – welche Funktionen soll das System anbieten, welche Informationen sollen wie auf Bildschirm-Masken aufgeteilt werden usw. – ausklammern. Das heisst, der Entwickler wird in seiner konkreten Projektarbeit auf Fragen stossen, die nur in Zusammenarbeit mit Benutzern, nicht aber durch Bücher beantwortet werden können.

Neben der besseren Anpassung der Software an Aufgaben und Unterstützungsbedürfnisse der Fachabteilung gibt es aber auch wirtschaftliche Gründe für den Einbezug von Benutzern. In einem Vergleich zwischen systemorientierten und benutzerorientierten Entwicklungsprojekten zeigten sich Vorteile bezüglich Wartungsaufwand, Kostenüberschreitung und Terminüberschreitung zugunsten einer Benutzerorientierung. Insbesondere die aktive Beteiligung von Benutzern in frühen Phasen der Entwicklung – bei der Problemanalyse und Anforderungsermittlung – kann zur Vermeidung nachträglicher, meist kostspieliger Korrekturen beitragen. Fehler in frühen Phasen bewirken Probleme in späteren Phasen und hohen Korrekturaufwand. Quantitative Analysen haben gezeigt, dass am meisten Fehler in der Problem- und Anforderungsanalyse entstehen, die dann auch den grössten Teil der Fehlerbehebungskosten verursachen.

2.1.3 Gegenseitiges Verständnis

Als grosser Vorteil sowohl für die Benutzer als auch für die Entwickler entpuppt sich der Gewinn an gegenseitigem Verständnis. Durch die Einsicht in die Entwicklungsarbeit begreifen die Benutzer die technischen Schwierigkeiten und Probleme, mit denen sich Entwickler bei der Realisierung von Benutzerwünschen häufig konfrontiert sehen, viel besser und akzeptieren auch eher eine abwehrende Haltung, wenn das Verhältnis von Aufwand zu Ertrag unverhältnismässig hoch ist. Umgekehrt staunt mancher Entwickler, wenn er die Unterschiede zwischen Stellenbeschreibungen, formalen Be-

schreibungen von Arbeitsabläufen usw. und dem realen Arbeitsverhalten durch die Zusammenarbeit mit Benutzern erfährt. Gewöhnlich sind Entwickler dann eher bereit, diese oder jene Funktion oder Änderung – die sie sonst als überflüssig erachtet hätten – noch zu realisieren, da sie deren Bedeutung für die Arbeit der Benutzer einsehen. Es finden auf beiden Seiten viele Aha-Erlebnisse statt!

Darüber hinaus gibt die Zusammenarbeit mit Benutzern dem Entwickler Feedback über die geleistete Arbeit, das über die Bemerkungen seines Vorgesetzten oder Testergebnisse hinausgeht. In vielen Fällen ist es das erste Mal, dass ein Entwickler derart detaillierte Rückmeldungen über Einsatz und Benutzung seines Arbeitsproduktes erhält, was auch sehr motivierend wirken kann.

Kasten 10: Beispiel für einen Benutzer(B)–Entwickler(E)-Dialog.

B: "Ich habe mich schon lange gefragt, warum man bei diesen Masken nur vorwärts-, aber nicht rückwärtsblättern kann. Das müsste doch einfach zu realisieren sein, und wäre für mich schon oft hilfreich gewesen!"

E: "Das liegt nicht an unseren Programmen, sondern an der Datenbank, auf der wir aufbauen, die kann das nicht, und wir müssten umfangreiche und komplizierte ..." (erläutert das Problem noch mehr).

B: "Aha, jetzt verstehe ich das, und ich dachte immer, ..., aber es ist nicht so schlimm, so wichtig ist das Rückwärtsblättern für mich auch wieder nicht, manchmal wäre es eben bequem!"

E: "Bei der nächsten Version der Datenbank, die wir ca. in einem halben Jahr bekommen, wird es dann möglich sein."

2.2 VORAUSSETZUNGEN FÜR BENUTZERBETEILIGUNG

2.2.1 Die erste Voraussetzung: Information

Eine der grundlegenden Voraussetzungen zur benutzerorientierten Planung und Durchführung organisationaler Veränderungen – wie z.B. der Entwicklung und Einführung neuer Anwendungssoftware – ist eine sachgemässe Information aller Betroffenen.

Warum

Fragmente und Details über geplante Projekte sickern – auch bei vertraulicher Behandlung von Dokumenten! – über informelle Kanäle, z.B. bei Pausengesprächen, zufälligen Treffen im Gang ("Hast Du schon gehört ...") usw. in den gesamten Betrieb, wo sie Unruhe erzeugen, Anlass zu Spekulationen geben und Keimzellen für Gerüchte bilden können.

Dies ist ein – vermutlich fast alltägliches – Beispiel für fehlende oder unzureichende Aufklärung über geplante Veränderungen. Gerüchte können aus Kleinigkeiten, unbestätigten Vermutungen, aufgeschnappten Bemerkungen usw. entstehen oder auch bewusst in die Welt gesetzt werden, z.B. um die 'Gegenposition' zu einer Stellungnahme zu provozieren. Ist ein Gerücht allerdings einmal entstanden, so entwickelt es eine Eigendynamik, die nur schwer zu stoppen ist.

Durch sachliche Information der betroffenen Mitarbeiter kann Ungewissheit reduziert und Spekulationen und Gerüchten vorgebeugt werden, wenn sie auch nicht immer ganz vermieden werden können! Verständliche Information von offiziellen Stellen – z.B. Projektleiter – mit Möglichkeiten der Rückfrage trägt zur Klärung offener Fragen bei und ermöglicht den Betroffenen auch eine bessere Abschätzung zukünftiger Entwicklungen; darüber hinaus hilft sie beim Aufbau eines vertrauenswürdigen Klimas im Betrieb.

In Deutschland ist die Information der Betroffenen gesetzlich geregelt; nach § 90 des Betriebsverfassungsgesetzes hat "der Arbeitgeber den Betriebsrat über die Planung ... 2. von technischen Anlagen, 3. von Arbeitsverfahren und Arbeitsabläufen oder 4. der Arbeitsplätze rechtzeitig zu unterrichten und die vorgesehenen Massnahmen insbesondere im Hinblick auf ihre Auswirkungen auf die Art der Arbeit und die Anforderungen an die Arbeitnehmer mit ihm zu beraten ..."

Wer

Wer soll überhaupt informiert werden? Selbstverständlich sind alle von einer Veränderung *direkt* Betroffenen zu informieren. Darüber hinaus kann es sich aber auch als sinnvoll erweisen, *indirekt* betroffene Mitarbeiter und Vorgesetzte anderer Abteilungen vom Vorhaben in Kenntnis zu setzen. Häufig können sich nämlich Veränderungen in den Arbeitsabläufen einer Abteilung auch ungeplant auf andere Abteilungen auswirken ("Seit ihr dieses neue System habt, kommen eure Dokumente viel schneller zu uns rüber; darauf waren wir gar nicht vorbereitet!"). Ausserdem kann es ratsam sein, über betrieblich vorgeschriebene Adressaten (wie Stabsstellen, Kommissionen, Standardisierungswesen usw.) hinaus auch Stellen zu informieren, die einen indirekten Einfluss auf das Projekt ausüben könnten.

Was

Worüber soll informiert werden? Die Mehrheit der Betroffenen interessiert sich in der Regel *nicht* so sehr für technische Aspekte wie Übertragungsraten, Speicherkapazitäten oder Zugriffszeiten von Computersystemen. Eine Ausnahme bilden heute z.T. qualifizierte PC-Benutzer. Informationen sollten deshalb vor allem über die (Hinter-)Gründe und Ziele des Vorhabens, die voraussehbaren Auswirkungen auf die Arbeitssituation und das weitere Vorgehen Klarheit schaffen. Den einzelnen Arbeitnehmer interessiert es zentral, wie sich seine Arbeitssituation, seine Rolle und seine Aufgaben, seine Kooperations- und Kommunikationsbeziehungen verändern werden. Kennt er noch den zeitlichen Horizont und die wichtigsten Schritte des Vorgehens, so kann er sich auch besser darauf einstellen.

Wie

Wie sollen die Informationen den Betroffenen übermittelt werden? Dazu gibt es verschiedene Möglichkeiten, deren Wahl grundsätzlich vom Umfang des Projektes und von der Anzahl der Betroffenen abhängt.

Für *kleinere* Projekte kann

- ein persönliches Gespräch,
- eine Orientierung am Anschlagbrett der Abteilung,
- ein persönliches Anschreiben,
- ein Zirkulationsschreiben in der Abteilung,
- eine Mitteilung in der Mailbox oder

- eine Orientierung durch den Vorgesetzten bei einer Sitzung

genügen. Für *grössere* Vorhaben lohnt es sich, einen dementsprechend grösseren Aufwand in die Information zu investieren, denn: Nachlässigkeiten und 'Sünden' zu Beginn eines Projektes führen zu einem schlechten Image, das sich nachträglich nur mit viel Mühe wieder aufbessern lässt!

Häufig benutzte Möglichkeiten sind

- Haus-, Betriebszeitung,
- Abteilungs-, Betriebsversammlung,
- Seminare, Workshops,
- Projektnachrichten,
- Projektbroschüre,
- Besuch eines anderen Betriebes oder eines Herstellers.

Letzteres ist eine viel zu selten praktizierte Möglichkeit. Indem man mit den Mitarbeitern einen Betrieb besucht, der eine ähnliche Veränderung schon durchgeführt hat, gibt man ihnen Gelegenheit, ein System im Alltagsbetrieb zu sehen und die Erfahrungen der Benutzer kennenzulernen.

Grundsätzlich ist darauf zu achten, dass die Information die Angesprochenen auch erreicht ("Was? Woher hast Du das? Ach, das steht in der Info-Box im Untermenü von 'Utilities'? Da schau' ich selten rein!"). In der Praxis werden deshalb auch meist mehrere der aufgelisteten Möglichkeiten kombiniert zur Verbreitung von Information genutzt. Für direkt Betroffene bzw. Beteiligte empfiehlt sich die Schaffung eines Verteilungsnetzes!

Wann

Wann soll die Information erfolgen? Diese Frage wird meist mit 'frühzeitig' beantwortet, weil in der Praxis die Information häufig zu spät erfolgt, die Zeitspanne aber auch schwerlich in Wochen oder Monaten angegeben werden kann. Sie hängt wiederum mit der Grösse des Vorhabens und den damit verbundenen betrieblichen Umwälzungen zusammen. Bei grösseren Projekten sind sechs Monate nach Projektetablierung eine untere Grenze, ein Jahr wird von den meisten Betroffenen als frühzeitig genug erachtet. Wenn sich eine Projektskizze konkretisiert und realisierbare Formen annimmt, wenn zuverlässige Informationen abgegeben werden können, ist meist auch bald der Zeitpunkt für eine Erstinformation gekommen. Dabei darf es aber nicht bleiben. Fortlaufende Information mit zunehmendem De-

taillierungsgrad soll die Betroffenen über den Stand der Arbeiten unterrichten! Wenn ein Projekt einmal begonnen wurde, ist es sinnvoll, die Betroffenen im Sinne eines Feedbacks periodisch über den Stand der Dinge, erzielte Erfolge ("Teilprojekt 'Automatische Agendierung' ist abgeschlossen. Die Anwendung wurde getestet und wird termingemäss verfügbar sein") wie auch Schwierigkeiten ("Bei der Verlegung der Kabel für das Netzwerk sind bauliche Probleme aufgetreten. Dadurch verzögert sich die Einführung des Systems um ca. drei Wochen") zu informieren.

Prinzipiell sollte bei jeder Information auch eine Person bzw. Kontaktstelle angegeben werden, die bei Unklarheiten oder weiterem Interesse der Benutzer Auskunft geben kann. Gegebenenfalls können auf diesem Wege auch schon erste Meinungen, Stellungnahmen zum Projekt eingeholt werden. Damit wird die Gefahr eines einseitigen Monologes vermieden und zum Dialog aufgefordert. Bezüglich Darstellung und Sprache ist auf Verständlichkeit der Information zu achten. Hinsichtlich der Menge an Information bzw. des Umfangs ist gerade bei Abgabe schriftlicher Informationen ein Mittelweg zwischen zu wenig Information und einer Überflutung mit Details zu finden, da sonst die Mitteilungen unverständlich bleiben oder nicht gelesen werden.

Letztlich bleibt es den Projektverantwortlichen überlassen, aufgrund von Erfahrung und Kenntnis der betrieblichen Gepflogenheiten aus den aufgelisteten Möglichkeiten ein der konkreten Situation angepasstes Informationskonzept zu entwickeln. Dabei sollte man allerdings nicht nur in 'Traditionen' verharren (" bei uns war das immer so ..."), sondern auch den Mut aufbringen, Neues zu wagen.

2.2.2 Die zweite Voraussetzung: Ausbildung

Als zweite wesentliche Voraussetzung für einen erfolgreichen Einbezug von Benutzern ist eine *angemessene Ausbildung* zu nennen. Es gibt zwei Qualifikationsarten: (1) die Qualifikation zur Teilnahme am Entwicklungsprozess und (2) die Qualifikation zur Benutzung des EDV-Systems. Im folgenden Abschnitt wird *nicht* über die Schulung aller Benutzer zur operationellen Handhabung des fertigen Systems gesprochen (siehe dazu Teil B, Kapitel 4.15), sondern speziell über die Qualifizierung der in die Entwicklung einbezogenen Benutzer(-vertreter).

Warum

Forderungen nach Benutzerbeteiligung wird manchmal mit dem allgemeinen Argument begegnet, die Benutzer hätten zuwenig Kenntnisse – z.B. in den Bereichen Organisation und Informatik – und könnten deshalb keine konstruktiven Beiträge leisten. Diesem Umstand kann durch eine entsprechende *Qualifizierung* der Benutzer, die sie *zu kompetenter Mitarbeit* befähigt, Rechnung getragen werden. Allerdings zeigen sich häufig auch bei Entwicklern Wissenslücken verschiedenster Art, weshalb im folgenden von *komplementärer Ausbildung* die Rede sein wird. *Die Qualifikationen von Benutzern und Entwicklern müssen komplementär erhöht und einander angenähert werden*, was auch zur Vermeidung bzw. Reduktion von Verständigungsproblemen führt.

Wer

Wie bereits angesprochen sollen nicht nur die beteiligten Benutzer, sondern auch die Entwickler qualifiziert werden. Grundsätzlich gilt, dass alle im Projekt involvierten Personen (Benutzer, Entwickler, Organisatoren, Koordinatoren usw.) – abhängig von ihrem Wissen und ihrer Erfahrung – eine Qualifikation erfahren sollen, die ihnen eine gemeinsame Basis vermittelt. Kombiniert mit dem jeweiligen Spezialwissen wird dadurch eine konstruktive Zusammenarbeit verschiedener Spezialisten gefördert.

Was

Sicher ist es für die Zusammenarbeit hilfreich, wenn den *Benutzern* ein *Grundlagenwissen in EDV* und *Kenntnisse* über die *Möglichkeiten und Grenzen* der aktuell verfügbaren *Technik* (z.B. "Was ist ein LAN? Wie sehen neuartige Benutzungsoberflächen mit windows, pull-down-Menüs aus" usw.) vermittelt wird. Genauso wichtig ist es aber, dass die *Entwickler* zumindest *Grundbegriffe des fachlichen Anwendungsbereiches* kennen. Wie sonst soll ein Entwickler eine Anwendung für eine Bank oder eine Versicherung erstellen, wenn er keine bankfachlichen oder versicherungstechnischen Kenntnisse hat? Um Missverständnisse zu vermeiden, sei aber festgehalten: Weder sollen Sachbearbeiter (als Benutzer) in einer Art 'Schnellbleiche' zu Programmierern noch sollen Entwickler zu Sachbearbeitern ausgebildet werden! Weiter hat es sich als sinnvoll erwiesen, wenn Benutzer und Entwickler Kenntnisse in *Organisation* und *(Software-)Ergonomie* erlangen. Darüber hinaus ist es speziell für Personen mit Vorgesetztenfunktion – z.B. Projektleiter und deren Stellvertreter – wichtig, dass sie Kenntnisse in *Gesprächsführung und Gruppenmoderation* erhalten. Da der

Arbeit in Teams grosse Bedeutung zukommt, ist eventuellen Qualifikationsdefiziten der Projektbeteiligten durch ein *Training sozialer Fertigkeiten und Kompetenzen* zu begegnen. Abhängig von den jeweiligen Aufgaben können sich zusätzliche spezifische Qualifikationserfordernisse (z.B. Präsentationstechnik, juristische Aspekte usw.) ergeben.

Wie

Prinzipiell ist der Erwerb der angesprochenen Inhalte in den üblichen betriebsinternen und -externen Kursen möglich. Für spezielle Inhalte – z.B. Gruppenmoderation, Gruppenarbeit – kann es sich jedoch als sinnvoll erweisen, einen auf die Projektbeteiligten und ihre konkrete Situation massgeschneiderten Kurs, Workshop zu organisieren. Meist sind es externe Experten, welche derartige Seminare durchführen. Selbststudium von Büchern und computergestütztes Lernen kann für die Aneignung bzw. Vertiefung von Grundwissen in einem Fachgebiet nützlich sein, darf aber interaktives Lernen in Gruppen nicht ersetzen!

Wann

Eigentlich, so möchte man meinen, ist es selbstverständlich, dass die Qualifizierung *vor* Beginn der Zusammenarbeit stattfindet. Praxiserfahrungen zeigen aber immer wieder, dass erst bei auftauchenden Problemen ad hoc improvisierte Massnahmen getroffen werden, was sich auf den Projektablauf störend auswirkt! Deshalb sei hier betont: *Vor* Beginn der eigentlichen Projektarbeit sollten Qualifikationsdefizite ermittelt und durch entsprechende Massnahmen ausgeglichen werden! Dies gilt im besonderen für die Anwendung neuer Entwicklungswerkzeuge, deren Erlernen und Beherrschung durch die Entwickler nicht in den Ablauf des Projektes verschoben werden sollte. Ebenso ist daran zu denken, dass Wissen und Kenntnisse in relativ kurzer Zeit erworben werden können, soziale Fertigkeiten aber über längere Zeit trainiert werden müssen, bis sie auch wirklich eingespielt sind.

2.3 KONZEPT DER BENUTZERBETEILIGUNG

Wie in einem Projekt Benutzerbeteiligung praktisch realisiert wird, hängt von einer Reihe von *Rahmenbedingungen* ab (siehe Teil B, Kapitel 2). Es sind dies z.B.:

(1) *Die Anzahl zukünftiger Benutzer:*

Bei nur 12 Benutzern ist eine *direkte* Beteiligung *aller* Benutzer noch gut möglich, mit steigender Benutzerzahl wird sich das Schwergewicht auf eine *indirekte* Beteiligung – die Mitwirkung von Benutzer*vertretern* – verschieben.

(2) *Der Projekttyp:*

Ob betriebsintern für eine Abteilung eine Individuallösung oder in einem Software-Haus ein Standard-Handelspaket entwickelt wird, bestimmt die Möglichkeiten des Zugangs zu Benutzern.

(3) *Die Komplexität der Anwendung:*

Je unstrukturierter, offener und komplexer die Aufgaben sind, desto unverzichtbarer ist eine aktive, kontinuierliche Mitarbeit von Benutzern.

Dies sind nur einige Rahmenbedingungen, die man berücksichtigen muss, bevor man das konkrete Beteiligungsmodell anhand der folgenden Kernelemente konzipiert.

2.3.1 Kernelemente eines Beteiligungsmodells

Form der Beteiligung

Wie erfolgt die organisatorische Eingliederung der Benutzer, wie wird die Zusammenarbeit zwischen Benutzern und Entwicklern geregelt: Aufbauorganisation, Aufgaben, Kompetenzen, Entscheidungsfindung?

Bei der Entwicklung *betriebsinterner* Anwendungen wird häufig ein Modell angewendet, das aus einem Lenkungsausschuss, einer Projektgruppe und Arbeitsgruppen besteht (siehe dazu Teil B, Kapitel 2). Bei der Entwicklung von Standardsoftware sind derartige Organisationsformen allerdings kaum möglich. Hier bietet sich z.B. die Bildung einer User-Group an, die sich in regelmässigen Abständen (mindestens einmal jährlich) mit den Entwicklern trifft, um z.B. Erweiterungen für neue Versionen eines Produktes zu diskutieren. Vielfach ist es in der Praxis auch der Fall, dass Pilotkunden Anregungen für neue Versionen liefern und diese dann – bevor sie in den Verkauf gelangen – einem Praxistest unterziehen. Viel zu wenig wird die Möglichkeit benutzt, über Verkäufer, Händler, Hotline oder Support ein Feedback über das Produkt zu erhalten. Eine systematisch geführte Dokumentation von Supportleistungen kann nicht nur auf Fehler in der Software hinweisen, sondern auch allgemeine Probleme der Benutzer im Umgang mit

der Software aufdecken und Verbesserungen anregen. Ferner besteht auch die Möglichkeit, über eine dem Produkt beigelegte Registrierkarte mit Fragebogen den Kontakt zu den ansonsten unbekannt bleibenden Benutzern herzustellen. Eine weitere Möglichkeit sind Umfragen in einschlägigen Fachzeitschriften.

Grad bzw. Ausprägung der Beteiligung

In welchem Ausmass werden die Benutzer in den Entwicklungsprozess einbezogen? Beschränkt sich die Beteiligung auf einen Informationsaustausch, werden die Benutzer(-vertreter) an Entscheidungen beteiligt und übernehmen sie auch Verantwortung und/oder wirken auch aktiv an der Gestaltungsarbeit (z.B. Maskendesign mit Hilfe eines Maskengenerators, bei der Datenmodellierung usw.) mit?

In einer Befragung von 157 Entwicklern, Benutzern und Führungskräften hat sich dazu ein interessanter Trend gezeigt: Alle drei Gruppen haben sich mehrheitlich dafür ausgesprochen, dass Benutzer nicht nur Informationen über ihre Aufgaben und Arbeitsabläufe liefern und Wünsche anbringen sollten, sondern vermehrt (Mit-)Entscheidungskompetenzen erhalten und Verantwortung übernehmen sollten. Teilweise – je nach Inhalt des Projektes – sollten Benutzer auch Teilaufgaben der Realisierung übernehmen, was wir auch in mehreren Projekten beobachten konnten! Dies ist ein Hinweis darauf, dass die bisher eher passive Rolle der Benutzer in einigen Unternehmen der Vergangenheit angehört und sich in Richtung einer Partnerschaft im Sinne der Kooperation von Experten verschiedener Gebiete entwickelt.

Inhalt der Beteiligung

An welchen Inhalten der Softwareentwicklung werden die Benutzer beteiligt?

In der Regel erstreckt sich die Mitarbeit auf Aufgaben, die mit der Systemfunktionalität, den Daten bzw. dem Datenkonzept und der Benutzungsoberfläche (Informationsdarstellung, Informationscodierung, Maskendesign, Dialogablauf) zu tun haben. Vorausgesetzt ist dabei, dass bei grösseren organisatorischen Veränderungen Fragen der Aufbau- und Ablauforganisation *vorher* mit den Benutzern geklärt wurden! Es wäre unsinnig, wenn Entwickler organisatorische Abläufe allein – ohne Benutzer – entwerfen und anschliessend für das Design von Masken die Benutzer einbeziehen!

Repräsentativität der Beteiligten

Werden alle, speziell ausgewählte oder repräsentative Vertreter der Benutzer einbezogen?

Eines der wichtigsten Kriterien der Benutzerbeteiligung ist die Repräsentativität! Dies ist eine Frage der 'richtigen' Zusammenstellung der Benutzerstichprobe, die in die Entwicklung einbezogen werden soll. In den meisten Fällen wird es kaum möglich sein, alle Benutzer kontinuierlich in die Entwicklungsarbeit zu integrieren. Die beteiligten Benutzer müssen aber repräsentativ für die ganze zukünftige Benutzerschaft sein. Je nach Projekt sind bei der Auswahl der beteiligten Benutzer folgende – unter Umständen auch noch andere! – Merkmale zu berücksichtigen, um Repräsentativität zu erreichen:

- Abteilungen, Fachbereiche,
- Aufgaben: Benutzer mit unterschiedlichen Aufgaben am System wie Eingabe, Bearbeitung, Abfrage,
- Benutzertypen: Benutzer mit unterschiedlicher Erfahrung und Benutzungshäufigkeit; oft können gerade Neulinge eine neue Sichtweise beisteuern; tägliche Benutzer haben andere Bedürfnisse als Benutzer, die einmal im Monat eine Abfrage durchführen,
- Hierarchiestufen,
- Sprachgruppen.

Methoden für Beteiligung

Welche Methoden (Instrumente, Werkzeuge, Tools) werden zu welchem Zeitpunkt im Entwicklungsprozess wofür (Analyse, Bewertung, Darstellung, Produktion, Test) eingesetzt?

Dies ist die Grundfrage der methodischen Gestaltung des Entwicklungsprozesses; sie sollte *vor* Projektbeginn beantwortet werden können! Das heisst, es ist zu überlegen, welche Methoden eingesetzt werden, ob sie vorhanden sind bzw. gekauft, gelernt, angepasst oder entwickelt werden müssen und ob sie eine spezielle Schulung verlangen oder das Vorhandensein bestimmter Geräte wie z.B. Videokamera und Rekorder bedingen! Müssen sie während des Projektes entwickelt oder gekauft und geschult werden, so ergeben sich unliebsame Verzögerungen im Ablauf und/oder improvisierte Methoden! Moderne Entwicklungswerkzeuge können vor allem in der Realisierungsphase den Programmieraufwand beträchtlich reduzieren. In der

Regel verlangen sie aber eine klar bestimmte Vorgehensweise, welche die Modellbildung und Entwurfsstrategie des Entwicklers – und damit auch seinen Handlungsspielraum – bestimmt. Unter Umständen können sie dadurch auch eine Benutzerbeteiligung erschweren.

Zeitpunkt der Beteiligung

Zu welchem Zeitpunkt, in welcher Phase im Projektablauf werden die Benutzer einbezogen?

Grundsätzlich sollen die Benutzer – abgesehen von den kontinuierlich mitarbeitenden Benutzervertretern – an *den* Arbeitsschritten beteiligt werden, wo sie auch einen konstruktiven Beitrag leisten können bzw. zu den Zeitpunkten, an denen Entscheidungen gefällt werden, die für die Benutzer bedeutsam sind. Wesentlich ist, dass der Einbezug *frühzeitig, das heisst schon bei der Problemanalyse, der Anforderungsermittlung und dem Grobkonzept erfolgt; es macht keinen Sinn, die Benutzer von diesen Aktivitäten auszuschliessen und sie dann bei der Detailspezifikation beizuziehen!* Normalerweise werden Benutzer bei der EDV-technischen Realisierung nicht einbezogen, da ihnen die fachlichen Kenntnisse fehlen. Bei Tests des – mehr oder weniger – fertigen Systems hingegen kann die Mitarbeit der Benutzer wichtig sein, wenn sich die Tests nicht nur auf programmiertechnische Aspekte beziehen, sondern auch den Umgang mit dem System enthalten. Letztlich wird die Beantwortung der Frage von der gewählten Beteiligungsform und den zur Diskussion stehenden Inhalten abhängig sein.

2.3.2 Evaluative Beteiligung

Die evaluative Beteiligung geht auf Mambrey und Oppermann (1983) zurück. Die Benutzer werden dabei zu definierten Zeitpunkten im Projekt – bei Vorliegen eines (Zwischen-)Ergebnisses, dessen Annahme oder Ablehnung über weitere Schritte entscheidet – zur Beurteilung und Bewertung von Vorschlägen, Prototypen, Systemmodulen einbezogen, die von den Entwicklern erarbeitet wurden. Diese eher passive Form der Beteiligung kann in unterschiedlicher Weise erfolgen (siehe Kasten 11).

Als *Vorteile evaluativer Beteiligung* sind zu nennen:

+ Die Entwickler erhalten ein Feedback und Input für die weitere Entwicklung, damit auch grössere Sicherheit, auf dem richtigen Weg zu sein.

+ Je nach Methode – z.B. bei Fragebogen – ergibt sich die Möglichkeit, eine grosse Zahl von zukünftigen Benutzern anzusprechen und damit ein repräsentatives Bild zu erhalten.

Kasten 11: Drei Beispiele für eine evaluative Beteiligung zur Beurteilung der Maskengestaltung.

(1) Die Vorschläge werden den Benutzern – auf Papier oder per EDV-Netz auf Terminal, PC – übermittelt. Mit einem Fragebogen wird die Beurteilung nach bestimmten Merkmalen erfragt und Verbesserungsvorschläge eingeholt ("Zeichnen Sie bitte selbst von Ihnen gewünschte Änderungen ein!"). Häufig werden dabei Alternativ-Vorschläge angeboten: "Welche Maske – A oder B – finden Sie besser?"

(2) In einem speziell einberufenen Workshop oder z.B. an einem Abteilungsmeeting werden die Vorschläge von den Entwicklern vorgestellt und mit den Benutzern diskutiert.

(3) Die Masken werden in einem Test mit einem Teil der Benutzer anhand realistischer Aufgaben geprüft. Dabei werden objektive Daten, z.B. wie schnell findet ein Benutzer die Information X auf Maske Y, evtl. im Vergleich mit Maske Z gemessen. Zusätzlich lassen sich subjektive Daten, z.B. Benutzerurteil, Einschätzung der Beanspruchung usw., erheben.

Als *Nachteile* sind zu werten:

– Die Beteiligung ist nur punktuell (zu bestimmten Zeitpunkten) und ausschnitthaft (zu bestimmten Aspekten). Aufgrund des organisatorischen Aufwandes wird eine derartige Beteiligung kaum mehr als zwei bis drei Mal in einem Projekt stattfinden. Die Benutzer können zeitlich und inhaltlich bedingte Zusammenhänge schwer erkennen; wichtige Bereiche werden ausgeklammert.

– Die Ergebnisse der Beteiligung 'verschwinden' bei den Entwicklern, man weiss nicht genau, was damit geschieht, und Umsetzungen werden erst viel später sichtbar und sind kaum mehr auf eigene Anregungen zurückzuführen ("Von meinen Vorschlägen kann ich nichts mehr

wiedererkennen; haben die das überhaupt zur Kenntnis genommen?"). Entsprechend ist die Motivation – und damit verbunden die Wahrscheinlichkeit kreativer Inputs – der Benutzer geringer, als wenn sie kontinuierlich in einen direkten interaktiven Austausch eingebunden sind.

2.3.3 Prozessuale Beteiligung

Die Benutzer bzw. ihre Vertreter arbeiten zu einem Teil ihrer Arbeitszeit kontinuierlich im Projekt mit und übernehmen dabei Teilaufgaben. Sie können nicht nur Stellung zu Vorgegebenem nehmen, sondern selbst bzw. in Zusammenarbeit mit Entwicklern Vorschläge entwerfen und dabei ihre Erfahrung konstruktiv umsetzen. Das Hauptgewicht dieser Art der Beteiligung liegt also auf aktiver, kontinuierlicher Mitarbeit bei der Entwicklung.

Kasten 12: Zwei Beispiele für eine prozessuale Beteiligung.

(1) Zwei Benutzer arbeiten zu 50% ihrer Arbeitszeit in der Projektgruppe. Sie wirken aktiv bei der Erarbeitung von Konzepten zum Funktionsumfang, zum Informationsgehalt und zur Benutzungsoberfläche mit; ausserdem ist es ihre Aufgabe, den Kontakt zu den übrigen 40 Benutzern sicherzustellen (Abgabe von Informationen, Einholen von Feedback, Organisation und Durchführung von Benutzer-Workshops).

(2) Vier Benutzer und ein Programmierer arbeiten in einer speziellen Arbeitsgruppe, die von der Projektgruppe den Auftrag hat, Masken, Formulare und Listen zu entwerfen und zu testen. Eine andere Arbeitsgruppe mit zwei Programmierern und einem Benutzer entwirft ein Konzept für die Benutzungsoberfläche und erstellt Prototypen.

Die zwei Beispiele in Kasten 12 zeigen, dass es auch für prozessuale Beteiligung unterschiedliche Organisationsformen geben kann.

Vorteile:

+ Durch die kontinuierliche Mitarbeit entwickeln die Benutzer(-vertreter) Einsicht in Zusammenhänge, was konstruktive, realistische Vorschläge und gegenseitiges Verständnis fördert.

+ Die Möglichkeit, von Grund auf eigene Vorschläge zu entwickeln und mit anderen Beteiligten zu diskutieren, wirkt motivierend.
+ Die Delegation von Teilaufgaben – wie z.B. Maskendesign, Kontakt mit übrigen Benutzern usw. – entlastet die Entwickler. Sie können sich vermehrt fachspezifischen Aufgaben (wie programmiertechnische Fragen, Schnittstellen zu anderen Systemen usw.) zuwenden.

Nachteile:

– Da die Benutzervertreter nur einen kleinen Teil aller zukünftigen Benutzer bilden, ist auch nicht immer gewährleistet, dass die übrigen Benutzer mit ihrer Arbeit bzw. Meinung übereinstimmen. Deshalb ist der ständige Kontakt zur gesamten Benutzerschaft wichtig!

Prozessuale und evaluative Beteiligung schliessen sich nicht aus, sondern lassen sich kombinieren: Die kontinuierliche, aktive Mitarbeit einiger Benutzervertreter wird zu definierten Zeitpunkten im Projekt durch die Stellungnahme aller oder zumindest eines repräsentativen Teils der Benutzer zu entscheidenden Fragen ergänzt!

Kasten 13: Gefahren bei indirekter Beteiligung: ein Beispiel.

Drei *Benutzervertreter* sammeln die Meinungen ihrer Arbeitskollegen zum vorgeschlagenen Maskendesign und übergeben sie einem *Koordinator;* dieser bereitet die 150 schriftlichen Beurteilungen auf, verdichtet sie, lässt – seiner Meinung nach – Unwichtiges weg und ergänzt Fehlendes; seine Interpretation des Haupttrends fügt er als Einleitung zum Bericht bei, den er dem *Projektleiter* übermittelt. Dieser liest vor allem die Einleitung sowie zusätzlich noch einige Abschnitte diagonal durch und übermittelt – befriedigt vom Gesamtergebnis in der Einleitung – den *Programmierern* einzelne Änderungshinweise. Nur schon ein grober Vergleich der originalen Benutzeraussagen mit den schliesslich getroffenen Änderungen ergab aber wesentliche qualitative und quantitative Diskrepanzen!

2.3.4 Direkte versus indirekte Beteiligung

Eine andere, grobe Unterscheidungsdimension betrifft die Frage, ob die Benutzer *direkt* (z.B. alle betroffenen Benutzer in einem Workshop mit

Entwicklern) oder *indirekt* (über Benutzervertreter, Koordinatoren, Betriebsrat usw.) in die Entwicklung einbezogen werden. Anders betrachtet geht es um den Weg, den Informationen vom Benutzer bis zum Entwickler zurücklegen müssen. Wie bei allen Informationsflüssen, die über mehrere Stellen verlaufen, besteht die Gefahr, dass die ursprünglichen Informationen gefiltert und verzerrt am Bestimmungsort ankommen. Aus zeitlichen, organisatorischen und ökonomischen Gründen bietet sich häufig der *indirekte Einbezug* an.

Die Auswahl der Benutzervertreter bestimmt massgeblich das Ergebnis der Mitarbeit; werden unmotivierte, schlecht qualifizierte, zur Zeit in der Fachabteilung gerade abkömmliche Benutzer beigezogen, so sind Probleme vorprogrammiert! Bei der Auswahl der Benutzervertreter ist deshalb – neben der Repräsentativität – sorgfältig auf deren Motivation, Qualifikation und soziale, kommunikative Fertigkeiten zu achten!

Die Möglichkeiten zur Benutzerbeteiligung sind vielfältig, aber es gibt kein Patentrezept. Vielmehr liegt es an den Verantwortlichen, ein auf ihr Projekt und ihren Betrieb massgeschneidertes Modell der Beteiligung zu konzipieren!

2.4 PRAXISPROBLEME BEI BENUTZERBETEILIGUNG

Es sei nicht verschwiegen: Auch bei noch so guter Planung können bei der Beteiligung von Benutzern Schwierigkeiten im Entwicklungsprozess auftreten. Nachfolgend sind deshalb einige der am häufigsten auftauchenden Probleme und mögliche Massnahmen zu ihrer Vermeidung genannt.

2.4.1 Doppelbelastung

Wenn ein Benutzer neben seinen täglichen Aufgaben *zusätzlich* noch als Benutzervertreter im Projekt Aufgaben übernehmen muss, so führt dies in der Regel zu einer Überforderung. Dies kann zur Folge haben, dass sein Engagement für das Projekt zunehmend sinkt, die Mitarbeit nur noch als Pflicht betrachtet und der Aufwand dafür minimal gehalten wird. Eine derartige Pflicht-Beteiligung ist für beide Seiten nicht nur nutzlos, sondern auch frustrierend! Deshalb ist es unbedingt notwendig, *Benutzervertreter* in Absprache mit ihrem jeweiligen Vorgesetzten offiziell zu einem gewissen Teil ihrer Arbeitszeit *von ihren täglichen Aufgaben zu entlasten.* Für einen Projektleiter erhöht sich die Gewissheit, dass er auf aktive Mitarbeit der Benutzervertreter zählen kann, da sie ja speziell dafür freigestellt wurden.

2.4.2 Fluktuation

Gerade bei länger dauernden Projekten kann die Fluktuation von Projektbeteiligten zu einem Problem werden, da mit den Personen ja nicht nur Wissen und Erfahrung verlorengehen, sondern auch eingespieltes Verhalten und zwischenmenschliche Beziehungen sich verändern. Sicherlich kann dieses Problem nicht restlos aus der Welt geschafft werden. Um einem ständigen Wechsel der Beteiligten vorzubeugen, sollten aber die *Bereitschaft und die objektiven Möglichkeiten bzw. Hindernisse* (bevorstehende Versetzung, Auslandaufenthalt, Weiterbildung usw.) für die Mitarbeit bis Projektende frühzeitig mit den Betreffenden *abgeklärt* werden. In Ausnahmefällen kann ein Personenwechsel auch Vorteile mit sich bringen (neue Ideen, weniger 'Betriebsblindheit' usw.).

2.4.3 Hierarchiegefälle

Bei hierarchisch gemischt zusammengesetzten Gremien (Projekt-, Arbeitsgruppen) können sich Probleme ergeben, wenn das alltägliche Rollenverhalten in der Gruppe beibehalten wird. Wenn Vorgesetzte dominieren und Unterstellte den Mut für Widerspruch und Kritik nicht aufbringen (dürfen?!), sind die Voraussetzungen für eine fruchtbare Zusammenarbeit, die auch unterschiedliche Meinungen zulässt, sehr schlecht. Neben der gemeinsamen Festlegung von *Regeln für Diskussionen und Verhalten in der Gruppe* kann sich ein *Training sozialer Fertigkeiten* für die Schaffung eines guten Gruppenklimas als nützlich erweisen.

2.4.4 Informationstransfer Beteiligte <–> Betroffene

Gerade bei länger dauernden Projekten kann sich die Gefahr ergeben, dass im Laufe der Zeit der Informationsfluss von den direkt Beteiligten (z.B. den Benutzervertretern) zu den übrigen Benutzern abnimmt oder gar versiegt. Die Gründe dafür können vielfältiger Natur sein, wie z.B. Nachlässigkeit, Vergessen, Überlastung, bewusste Zurückhaltung usw. Meist nimmt dabei aber auch der Informationsfluss in umgekehrter Richtung – von den Betroffenen zu den Beteiligten – ab, was dem Projektfortschritt abträglich ist. In Extremfällen kann es sogar zu einer Abschottung der Projektbeteiligten von den übrigen Betroffenen kommen, was einerseits zu sozialen Spannungen bzw. Konflikten und andererseits zu einer einseitigen Lösungsfindung mit entsprechend eingeschränkter Akzeptanz führen kann. Um derartige Entwicklungen zu vermeiden, ist die *Institutionalisierung des Informationsaustausches,* z.B. in Form *regelmässiger Meetings* oder dem

Aufbau eines *Informationsverteilungsnetzes* (siehe auch Teil A, Kapitel 2.2.1), erforderlich.

2.4.5 Fixierung, Struktur-Konservatismus

Ein allgemeines Problem besteht häufig darin, dass Projektgruppen sich allzu sehr an bestehenden Strukturen oder organisationalen Abläufen orientieren und der innovative Gehalt einer Lösung entsprechend bescheiden ausfällt. Oft passiert es auch, dass eine Gruppe sich zu früh auf eine erste sich ergebende oder die nächstliegende Lösung konzentriert bzw. fixiert, ohne Alternativen wirklich geprüft zu haben. Dieser Gefahr kann durch den Einsatz von Kreativitätstechniken und die Gewährung eines auch für anfänglich abwegig erscheinende Ideen *offenen Diskussionsklimas* in der Gruppe begegnet werden. Auch die Einführung einer 'Spielverderber'-Rolle im Sinne eines Advocatus Diaboli kann eine vorschnelle Einigung auf eine suboptimale Lösung verhindern helfen.

2.4.6 Motivationserhalt

Grosse, längerfristige Projekte haben häufig den Nachteil, dass Ergebnisse der Projektarbeit erst nach einigen Monaten oder gar Jahren sichtbar werden. Ein derart verzögertes Feedback dämpft aber die Motivation der Benutzer(-vertreter) und bewirkt nachlassendes Engagement und Initiative. Deshalb ist es sinnvoll, grosse Projekte in *kleinere, überschaubare Teilprojekte* aufzugliedern, deren Abwicklung den Beteiligten *sichtbaren Fortschritt und Erfolgserlebnisse* bringt.

2.4.7 Vorstellungsvermögen der Benutzer bzw. ihrer Vertreter

Die in der Softwareentwicklung häufig eingesetzten Methoden (formale Spezifikationen, Diagramme, Struktogramme, Pflichtenhefte usw.) überfordern in der Regel das Vorstellungsvermögen von Benutzervertretern. Anhand dieser Unterlagen können sie sich kaum eine gedankliche Vorstellung des zukünftigen Systems bilden. Im Gegensatz dazu sind *Prototypen anschauliche Modelle* für die Beurteilung und Ableitung von Verbesserungs- bzw. Änderungsvorschlägen seitens der Benutzer.

2.4.8 Verständigung zwischen Benutzern und Entwicklern

Eines der Hauptprobleme ist die Kommunikation zwischen Benutzern und Entwicklern. Vielfach bereitet es den beiden Personengruppen einige

Mühe, das Gemeinte dem jeweils anderen auch verständlich zu übermitteln. Unverständnis bzw. Missverständnisse sind die Folgen, die sich – bei anhaltenden Schwierigkeiten – zu Rückzug und gegenseitiger Ablehnung der beiden Gruppen ausweiten können. Manchmal stellt sich auch nachträglich heraus, dass unter dem gleichen Begriff Verschiedenes verstanden wurde. Verständigungsprobleme resultieren nicht nur aus dem unterschiedlichen Sprachgebrauch. Vielmehr sind sie Ausdruck für unterschiedliche

(1) Ausbildungen,

(2) Betrachtungsperspektiven des zu entwickelnden Systems und

(3) Denkstile bzw. Logik beim Problemlösen.

Entwickler denken meist in abstrakt-analytischen, systemanforderungsbezogenen Kategorien, Benutzer hingegen in aufgaben- und anwendungsbezogenen Begriffen.

Kasten 14: Beispiel einer unzureichenden Benutzer(B)–Entwickler(E)-Verständigung.

B: "Ich muss neue Bücher eingeben, verschlüsseln und ..."
E: "Ja, Schlüsselmerkmale müssen für Selektionen definiert werden."

B: "Ääh ... ich meine thematisch ...ja, und katalogisieren, Schlüssel ändern..."
E: "Mmh, nicht so einfach bei dieser relationalen DB."

B: "Ja...? Dann nach Autoren und Jahrgang sortieren, auswählen und verschiedene Listen erstellen und ausdrucken."
E: "Ja, dann machen wir ein Menü 'Erfassung, Mutation, Anzeigen', sauber getrennt, damit bei 'Anzeigen' auch nichts geändert werden kann."

B: "Aber ich muss doch Angezeigtes ändern können..."
E: "Ja,ja, dann ein Menü 'Selektion', wo Sie auch nach Autor oder Jahrgang sortieren können."

B: "... und Jahrgang, nach beidem!"
E: "Das geht bei dieser DB nicht! Wieviele Records haben Sie überhaupt?"

B: " Records ...?"
usw.

Derartige Verständigungsprobleme können durch *komplementäre Ausbildung* (siehe Teil A, Kapitel 2.2.2) und die Schaffung eines *gemeinsamen Begriffsglossars* verringert werden. Sie entschärfen sich auch zunehmend, da jüngere Benutzergenerationen schon in der Schulausbildung EDV-Kenntnisse erhalten. Weniger Kommunikationsprobleme treten auf, wenn Entwickler eine fachspezifische Vorbildung (z.B. kaufmännische Grundausbildung) besitzen!

2.4.9 Abwehr

Ein Problem kann sich dadurch ergeben, dass Benutzer zur Einbringung von Vorschlägen bzw. Anforderungen aufgefordert werden, eintreffende Vorschläge dann aber von Entwicklerseite pauschal ("... das geht nicht")

abgewehrt werden. Ob diese Abwehr nun gerechtfertigt ist (Aufwand, technische Restriktionen) oder nicht: Die Motivation der Benutzer wird dadurch zusehends reduziert und kippt möglicherweise in Resignation oder Aggression über. Es ist deshalb wichtig, schon zu Projektbeginn die *Erwartungen* auf einem *realistischen Niveau* zu halten und *frühzeitig* über *mögliche Restriktionen* zu *diskutieren*. Die Strategie, die Benutzer durch grosse Versprechen für das Projekt zu gewinnen und die Versprechen dann nicht zu erfüllen, kann sich als fatal erweisen!

2.4.10 Inkonsistenz von Benutzeranforderungen

Von Entwicklern wird häufig die Inkonsistenz von Benutzeranforderungen beklagt ("... vor drei Wochen wollten sie es so, nun sieht's wieder anders aus ...!"). Diese für Entwickler wirklich unangenehme Situation lässt sich nicht grundlegend beseitigen. Bei der Komplexität heutiger Anwendungen wäre die Annahme unrealistisch, dass alle zu Beginn ermittelten Anforderungen über die Projektlaufzeit hinweg konstant blieben. Eine Entschärfung des Problems ergibt sich durch eine *schrittweise Präzisierung und Differenzierung von Anforderungen z.B. durch Prototyping* (siehe Teil B, Kapitel 4.10).

Diese Problemauflistung soll niemanden entmutigen oder gar von einer Benutzerbeteiligung abhalten! Probleme werden in den meisten Projekten auftauchen, können aber besser gelöst werden, wenn man sie antizipiert. Mit zunehmender Erfahrung werden sie abnehmen, und Benutzerbeteiligung wird fester, 'normaler' Bestandteil der Softwareentwicklung werden.

2.5 FAZIT

Die Ausführungen in diesem Kapitel zeigen, dass es verschiedene, gute Gründe für die Beteiligung von Benutzern gibt. Die grundlegendste und vielleicht wichtigste Voraussetzung ist, dass die Benutzer frühzeitig und ausreichend informiert und ausgebildet werden, bevor ein auf das konkrete Projekt abgestimmtes Beteiligungskonzept gemeinsam entwickelt wird. Die aufgelisteten Probleme bei Beteiligungsprozessen sollen zeigen, dass auch Benutzerbeteiligung ein Lernprozess ist, der nur gemeinsam bewältigt werden kann.

TEIL B: ORGANISATION DER SOFTWARE-ENTWICKLUNG

1 Einleitung

Softwareprojekte sind zunehmend durch komplexe, innovative und zeitkritische Problemstellungen gekennzeichnet. Das Projektmanagement dient dazu, diese schwierige Situation zu bewŠltigen, und umfasst die fŸr ein Projekt notwendigen planenden, steuernden, Ÿberwachenden, methodischen und personalbezogenen AktivitŠten zur Sicherstellung des Projekterfolges.

Zur Erfolgsmessung sind vor allem folgende Kriterien von Relevanz:

(1) Die Leistung und QualitŠt des Systems,

(2) der Kosten- und Zeitaufwand fŸr die Systementwicklung,

(3) die Zufriedenheit der Projektbeteiligten und -betroffenen mit dem Projektverlauf und dessen Ergebnis,

(4) die Qualifizierungsmšglichkeiten der Projektbeteiligten im Rahmen des Projektes,

(5) die UnterstŸtzung der Benutzer durch das System,

**Rauterberg, M., Spinas, P., Strohm, O., Ulich, E. & Waeber, D. (1994).
Benutzerorientierte Software-Entwicklung Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung.
(Mensch - Technik - Organisation. Band 3), vdf Hochschulverlag AG an der ETH Zurich.**

(6) der betriebswirtschaftliche Nutzen des Systems im Anwendungskontext.

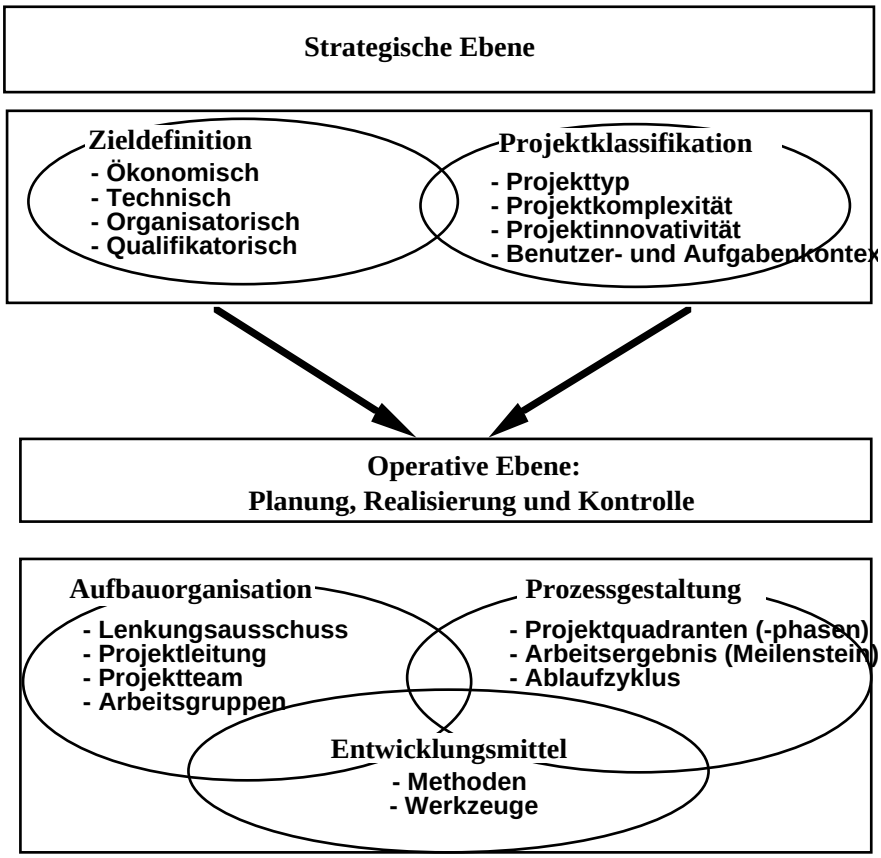


Abbildung 4: Elemente des Projektmanagements.

Einige der genannten Kriterien stehen in unmittelbarem Zusammenhang zueinander. So ist die Zufriedenheit der Benutzer sicherlich eine notwendige Bedingung für den betriebswirtschaftlichen Nutzen des Systems. Eine problemangepasste, aufgaben- und benutzerorientierte Gestaltung des Projektmanagements bildet die wesentliche Voraussetzung für einen an diesen Kriterien bemessenen Projekterfolg. Für Softwareentwicklungsprozesse sind

vor allem die folgenden Elemente des Projektmanagements von Bedeutung (vgl. Abbildung 4):

(1) Strategische Ebene: Zieldefinition und Projektklassifikation.

(2) Operative Ebene: Planung, Realisierung und Kontrolle, Aufbauorganisation, Prozessgestaltung und Entwicklungsmittel.

Abbildung 4 zeigt, dass die einzelnen Elemente des Projektmanagements nicht isoliert nebeneinanderstehen, sondern in vielfältiger Weise logisch, operativ und zeitlich miteinander verbunden sind. Dies bedeutet konsequenterweise, dass die sorgfältige Bearbeitung der strategischen Projektschritte *'Zieldefinition'* und *'Projektklassifikation'* eine zentrale Voraussetzung für die angemessene Planung, Realisierung und Kontrolle des Projektes auf operativer Ebene ist. Fehler und/oder Unklarheiten in den ersten Projektschritten führen zwangsläufig zu einer fehlerhaften und/oder unangepassten Projektabwicklung.

Die Prinzipien des *soziotechnischen Systemkonzeptes* bilden eine nützliche Grundlage für eine ganzheitliche Gestaltung von Softwareprojekten. In Teil A, Kapitel 1.1 wurde das soziotechnische Systemkonzept vorgestellt, demzufolge Arbeitssysteme aus einem *sozialen* und einem *technischen Teilsystem* bestehen. Für die Gestaltung effizienter Arbeitssysteme ist von entscheidender Bedeutung, beide Systeme gemeinsam zu optimieren und aufeinander abzustimmen.

Eine Übertragung dieser Sichtweise auf Softwareprojekte bedeutet:

Das *soziale Teilsystem* ist durch die Projektbeteiligten mit ihren projektspezifischen Bedürfnissen, Interessen und Qualifikationen gekennzeichnet.

Das *technische Teilsystem* umfasst u.a. mögliche Methoden und Werkzeuge zur Erfüllung der Projektaufgaben auf strategischer und operativer Ebene.

Für die konkrete Gestaltung der Aufbau- und Ablauforganisation sowie die definitive Bestimmung der Entwicklungsmittel bestehen – in Abhängigkeit von den Projektzielen und der Projektklassifikation – Wahlmöglichkeiten, die im Sinne der Gestaltungskriterien des soziotechnischen Systemkonzeptes sowie einer interessen- und bedürfnisausgleichenden Berücksichtigung der verschiedenen Projektbeteiligten und -betroffenen genutzt werden sollten.

Hinsichtlich der Aufbauorganisation eines Projektes sind diese Wahlmöglichkeiten z.B. mit folgenden Fragen verbunden: Aus welchen Personen setzt sich die Projektgruppe zusammen? Welche Rolle nehmen die verschiedenen Projektbeteiligten ein? Welche Kompetenzen haben diese Projektbeteiligten? Liefern die Benutzer lediglich fachspezifische Informationen und evaluieren ad hoc die Lösungsvorschläge der Entwickler oder arbeiten sie

aktiv an der Konzeption und Realisierung des Systems in der Projektgruppe mit? Werden neben der Projektgruppe themenspezifische Arbeitsgruppen eingerichtet?

In einem hochkomplexen und innovativen Projekt z.B. wäre im Sinne einer soziotechnischen Optimierung die Integration von Benutzern in gemischten Projekt- und Arbeitsgruppen auch mit Implikationen für die Gestaltung anderer Elemente des Projektmanagements verbunden. So wäre ein iterativ-zyklisches Vorgehen – unter Einsatz von Prototyping oder Versioning – ein adäquates Pendant zu einer solchen Aufbauorganisation. Diese Form der zyklischen Ablauforganisation wird wiederum durch den Einsatz von Entwicklungsmitteln wie Aufgabenanalysen, Werkzeuge wie Masken- und Dialoggeneratoren, Programmiersprachen der vierten Generation usw. am besten unterstützt bzw. erst praktikabel. Spätestens zu diesem Zeitpunkt kommen Fragen bezüglich Investitionen in die Entwicklungsumgebung und Qualifizierungsmassnahmen für die Projektarbeit auf.

Diese Fragen und Implikationen zeigen die Notwendigkeit und auch Vorteile einer soziotechnischen bzw. ganzheitlichen Betrachtung bei der Planung und Gestaltung von Softwareprojekten. Diese ganzheitliche Sichtweise hilft die einzelnen Elemente des Projektmanagements gemeinsam zu optimieren und aufeinander abzustimmen. Die Ergebnisse der Zieldefinition und Projektklassifikation auf strategischer Ebene bilden dafür eine zentrale Voraussetzung.

2 Projektorganisation

2.1 DIE ZIELDEFINITION

In der betrieblichen Softwareentwicklungspraxis besteht zu Projektbeginn haufig das Problem, dass die Projektziele unklar bleiben bzw. unzureichend definiert werden. Dies fuhrt zu mangelnder Zielorientierung und inadŠquater Losungssuche, was sich in falschen Entscheidungen und Losungen niederschlagen kann. Eine gewisse Klarheit bez. ubergeordneter Projektziele, welche in der Regel okonomischer, technischer, organisatorischer und qualifikatorischer Natur sind, sollte daher vor der Realisierung bestehen. Im Rahmen eines solchen Zielbildungsprozesses sind folgende Schritte zu durchlaufen:

(1) Zielideen generieren,

(2) Zielkatalog formulieren,

(3) Ziele gewichten,

(5) Zielbeziehungen analysieren,

(6) Zielkonflikte bereinigen,

(7) Ziele operationalisieren,

(8) Zieldefinition formulieren.

Eine schriftliche Dokumentation der Zieldefinition ist eine wesentliche Voraussetzung zur Durchsetzung der Ziele und liefert eine wichtige Grundlage fur die Gestaltung des Projektes auf operativer Ebene. Eine konsensorientierte, interessenausgleichende Zieldefinition in technisch-organisatorischer und betriebswirtschaftlicher Hinsicht zwischen Geschaftsleitung, EDV- und Fachbereich ist dabei als eine der zentralen Voraussetzungen fur den Projekterfolg anzustreben.

Damit die formulierten Ziele immer auf einem aktuellen Stand gehalten werden konnen, mŸssen im Laufe des weiteren Prozesses Kontrollen durchgefuhrt werden, da z.B. Anderungen im Geschaftsumfeld oder im technischen Umfeld Ð welche vor allem bei grossen Projekten mit entsprechender Laufzeit wahrscheinlich sind Ð die Modifikation der formulierten Ziele erfordern. Diese Aktualisierung sollte im Rahmen solcher Kontrollschritte geleistet werden.

Rauterberg, M., Spinas, P., Strohm, O., Ulich, E. & Waeber, D. (1994).
Benutzerorientierte Software-Entwicklung Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung.
(Mensch - Technik - Organisation. Band 3), vdf Hochschulverlag AG an der ETH Zurich.

Ein iterativ-zyklisches Vorgehen bei der Softwareentwicklung ist auf die Änderungsdynamik von Zielen und Anforderungen ausgerichtet. Dies ist eine Notwendigkeit, damit solche Zielmodifikationen auch umgesetzt bzw. realisiert werden können.

2.2 DIE PROJEKTKLASSIFIKATION

Eine oberflächliche Analyse von Projektrahmenbedingungen bzw. eine unsystematische Projektklassifikation führt zur Abwicklung von 'Einheitsprojekten', wobei unterschiedlichen Ausgangslagen verschiedener Projekte nicht adäquat Rechnung getragen wird. Dies kann zur Folge haben, dass Ziele nicht erreicht werden oder – im schlechtesten Fall – ein Projektabbruch nötig wird. Im folgenden Abschnitt werden wesentliche Rahmenbedingungen bei der Softwareentwicklung beschrieben, die der Projektklassifikation dienen.

Für ein aufgaben- und benutzerorientiertes Projektmanagement sind folgende Rahmenbedingungen von besonderer Wichtigkeit.

2.2.1 Der Projekttyp

Vier charakteristische Projekttypen sind zu unterscheiden:

Typ A: Entwicklung einer Individuallösung für eine firmeninterne Fachabteilung.

Typ B: Entwicklung einer Individuallösung für einen externen Kunden.

Typ C: Entwicklung einer Branchenlösung für einen weitgehend bekannten Kundenkreis.

Typ D: Entwicklung einer Standardlösung für einen breiten und anonymen Benutzerkreis.

Immer häufiger werden auch Projekte abgewickelt, bei denen branchenspezifische Softwarelösungen ausgewählt und durch die interne Entwicklungsabteilung oder eine externe Softwarefirma bzw. den Softwarelieferanten firmenspezifisch angepasst werden.

Die verschiedenen Projekttypen unterscheiden sich erheblich in den Möglichkeiten zur Aufgaben- und Benutzerorientierung bei der Softwareentwicklung, da von Projekttyp A zu D

(1) die Bekanntheit der Benutzer,

(2) die Bekanntheit der zu unterstützenden Aufgaben,

(3) die Bekanntheit des organisatorischen Kontextes und

(4) die Zugangsmöglichkeiten zu den Benutzern

immer weniger gegeben sind. Die raumzeitliche Nähe zwischen Benutzern und Softwareentwicklern wie auch die infrastrukturellen Voraussetzungen für deren Zusammenarbeit sind z.B. in Abhängigkeit vom Projekttyp sehr unterschiedlich. Entsprechend sind verschiedene Formen der Projektorganisation und Zusammenarbeit zu praktizieren. Unterschiedliche Varianten der Projektorganisation werden daher in Teil B, Kapitel 2.3 beschrieben.

2.2.2 Die Projektkomplexität

Die Komplexität von Softwareprojekten wird u.a. durch folgende Merkmale bestimmt:

- Umfang der geplanten Systemfunktionalität,
- Anzahl und Unterschiedlichkeit der betroffenen Fachaufgaben (siehe dazu auch Benutzer- und Aufgabenkontext),
- Umfang der geplanten arbeitsorganisatorischen Veränderungen für die Fachbereiche bzw. Anwendungsbereiche,
- Umfang der geplanten qualifikatorischen Massnahmen für die potentiellen Systembenutzer.

Anhand dieser Merkmale eingestuft, ist etwa die Entwicklung eines Lieferantenverwaltungssystems in einer Einkaufsabteilung als ein Projekt von eher geringer Komplexität zu betrachten. Die Entwicklung eines integrierten Produktionsplanungs- und -steuerungssystems, das sowohl die informationstechnische wie auch die organisatorische Integration der Bereiche Verkauf, Disposition, Einkauf und Fertigung zum Gegenstand hat, zeichnet sich dagegen durch eine sehr hohe Komplexität aus. Komplexe Projekte sind in der Regel durch eine lange Projektdauer, einen grossen Personenaufwand sowie eine grosse Anzahl von Projektbeteiligten gekennzeichnet.

2.2.3 Die Projektinnovativität

Eine wesentliche Rahmenbedingung ist die Innovativität bzw. Neuartigkeit eines Projektes. Innovative Projekte sind – im Vergleich zu Routineprojekten – durch eines oder mehrere der folgenden Merkmale gekennzeichnet:

(1) Das System ist eine Neu- oder Erstentwicklung,

(2) in der Entwicklungsabteilung bestehen keine Erfahrungen mit diesem Fachbereich, dieser Branche bzw. der Art der zu unterstützenden Aufgaben,

(3) in der Entwicklungsabteilung bestehen keine Erfahrungen mit dieser Art von Anwendungen, Automatisierung, Techniklösung,

(4) im Fach- bzw. Anwendungsbereich besteht eine geringe EDV-Unterstützung und -Erfahrung.

Bei innovativen Projekten ist also das Zurückgreifen auf Erfahrungswerte nur bedingt möglich. Die Planung solcher Projekte ist deshalb erschwert. Vor diesem Hintergrund setzen innovative Projekte umfassende Informationen über den Fachbereich (benötigte Daten und Funktionen, Arbeitsabläufe usw.) voraus. Umfassende Analysen sowie konzeptionelle Arbeiten mit den Benutzern zusammen (zur Nutzung ihres Fachwissens!) bilden eine wesentliche Voraussetzung für den Erfolg von innovativen Softwareprojekten.

2.2.4 Der Benutzer- und Aufgabenkontext

Anwendungssoftware wird zur Unterstützung von sehr unterschiedlichen Benutzergruppen und Aufgabentypen entwickelt. Benutzer- und aufgabenorientierte Softwareentwicklung erfordert daher, dass dieser Unterschiedlichkeit beim Vorgehen Rechnung getragen wird. Die Unterschiedlichkeit von Benutzergruppen zeigt sich vor allem an folgenden Merkmalen:

(1) Anzahl der potentiellen Benutzer,

(2) fachlicher bzw. qualifikatorischer Hintergrund der Benutzer,

(3) hierarchische Ebene der Benutzer,

(4) EDV-Erfahrung der Benutzer.

Aufgaben im Büro- und Verwaltungsbereich unterscheiden sich u.a. in den folgenden Merkmalen:

(1) Strukturiertheit, Formalisierbarkeit der Aufgaben,

(2) Vielfalt und Variabilität der (Teil-)Aufgaben,

(3) Kommunikations- und Kooperationserfordernisse,

(4) Umfang der Freiheitsgrade im Aufgabenvollzug,

(5) Struktur und Menge erforderlicher Informationen, Funktionen,

(6) Stellenwert der EDV bei der Aufgabenerfüllung.

Kasten 15: Projekt Kundenverwaltung.

Die interne Entwicklungsabteilung einer Kleinbank ist damit beauftragt, ein neues System zur Kundenverwaltung zu entwickeln. Das alte funktionstastengesteuerte System, das auf Stand-alone-PCs implementiert ist, soll durch ein System mit Window-Oberfläche und leicht erweiterter Funktionalität ersetzt werden; zudem sollen die Stand-alone-PCs miteinander vernetzt werden. Die Arbeitsorganisation der betroffenen Fachabteilung – in der fünf Teilzeitmitarbeiterinnen mit der Erfassung und Pflege von Kundenstammdaten beauftragt sind – soll im Rahmen des Projektes weitgehend unverändert bleiben.

Kasten 16: Projekt Bibliothek.

Eine Softwarefirma ist damit beauftragt, ein integriertes Bibliothekssystem für eine Universität zu entwickeln. Der Bibliotheksbetrieb wurde bisher weitgehend konventionell mit Mikrofichen, Katalogen, Bestellformularen usw. abgewickelt. Mit Hilfe eines interaktiven On-line-Systems sollen nun eine komfortable Anlage und Pflege von Bibliotheksdatensätzen, komplexe Abfragemöglichkeiten, die Vorbestellung von Büchern usw. universitätsintern und dezentral in externen Instituten mittels eines Netzwerkes ermöglicht werden.

Potentielle Systembenutzer sind das Bibliothekspersonal, Sekretärinnen, wissenschaftliche Mitarbeiter, Dozenten, Studierende usw. Das Bibliothekspersonal setzt sich aus gelernten Bibliothekaren, Sachbearbeiterinnen sowie studentischen Hilfskräften zusammen. Die bisher praktizierte Arbeitsteilung in der Bibliothek soll bei der Systemkonzeption neu überdacht werden. Es ist vorgesehen, das System auf dem universitätsinternen Grossrechner zu implementieren. Für das wissenschaftliche Personal soll dabei die Möglichkeit bestehen, mittels den an ihren Arbeitsplätzen installierten PCs Zugriff auf das zukünftige Bibliothekssystem zu haben.

Der Stellenwert des EDV-Systems bei der Aufgabenerfüllung zeigt sich u.a. daran, ob ein System als integrales, zentrales Hilfsmittel für den Grossteil der Aufgaben oder nur zur Unterstützung einzelner Aufgaben benötigt wird.

Aus den beschriebenen Rahmenbedingungen wird deutlich, dass die Ausgangslage bei Softwareprojekten sehr unterschiedlich sein kann. An zwei Projektvorhaben (Projekt Kundenverwaltung und Projekt Bibliothek) wird dies veranschaulicht (vgl. Kasten 15 und Kasten 16). Entsprechend den Projektbeschreibungen lassen sich die in Kasten 15 und 16 dargestellten Projekte wie folgt klassifizieren:

Rahmenbedingungen	Projekt Kundenverwaltung	Projekt Bibliothek
Projekttyp	A	B
Projektkomplexität	gering	hoch
Projektinnovativität	gering	hoch
Benutzergruppen	homogen	heterogen
Aufgaben	einfach, Routine	vielfältig, anspruchsvoll

Grundsätzlich ist bei der Zieldefinition und Projektklassifikation zu beachten, dass diese Projektschritte in einem Wechselverhältnis stehen (Massnahmen der Aufgabenerweiterung als arbeitsorganisatorische Projektziele erhöhen z.B. auch die Vielfältigkeit und Variabilität der betroffenen Fachaufgaben). Daher sollten diese Projektschritte aufeinander abgestimmt und unter Zusammenarbeit der Auftraggeber und Auftragnehmer sowie der Projektbeteiligten aus den Fach- und EDV-Bereichen erfolgen.

2.3 DIE AUFBAUORGANISATION

Die Aufbauorganisation regelt u.a. die Informations- und Entscheidungswege, Verantwortlichkeiten sowie Arbeitsteilung in einem Projekt. Dabei werden häufig strategische, steuernde sowie operative Gremien bzw. Funktionsträger unterschieden. Durch die Aufbauorganisation ist u.a. festgelegt, welche Personen bzw. Funktionsträger welchem Gremium angehören und welche Aufträge bzw. Aufgaben im Verantwortungsbereich der einzelnen Gremien liegen. Wesentliche Kriterien für die Bewertung der Aufbauorganisation sind u.a. die Zentralität vs. Dezentralität, die Möglich-

keiten zur Mitwirkung der Projektbeteiligten und -betroffenen sowie das Strukturierungsprinzip im Sinne funktionaler Arbeitsteilung vs. funktionale Integration (vgl. Ulich 1994).

2.3.1 Die traditionelle Aufbauorganisation

Traditionelle Modelle der Aufbauorganisation von Softwareprojekten entsprechen der in Abbildung 5 dargestellten Struktur. Aus der Darstellung in Abbildung 5 lassen sich eine Reihe von Hauptmerkmalen und Problemen traditioneller Strukturen bei der Softwareentwicklung ableiten.

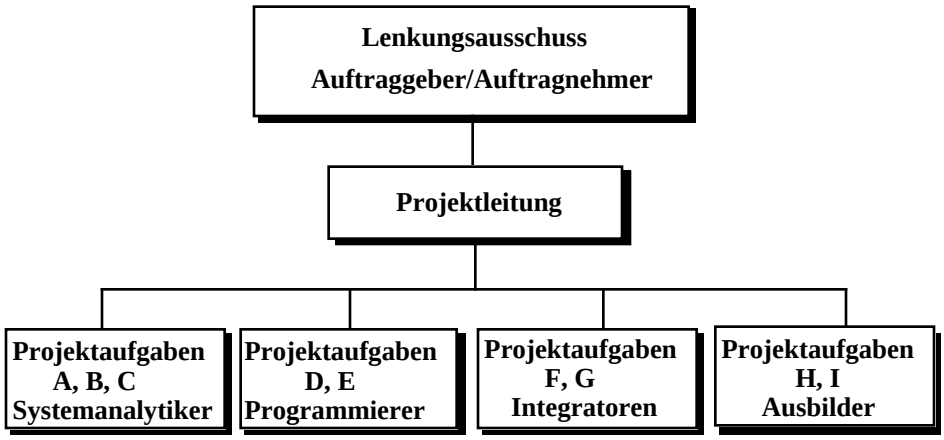


Abbildung 5: Vereinfachte Darstellung eines traditionellen Modells einer Aufbauorganisation.

1. *Die Zielvorgabe, Terminierung und Budgetierung erfolgt zentral, z.B. vom Lenkungsausschuss, das heisst, ohne Einflussmöglichkeiten der am Entwicklungsprozess direkt Beteiligten und Projektmitarbeiter auf operativer Ebene.*

Damit kann ein grosser Kosten- und vor allem auch Termindruck für die Projektmitarbeiter auf operativer Ebene verbunden sein. Starker Termindruck führt häufig zu Kreativitätshemmnissen, Überbeanspruchung und sogar Überforderung bei den Projektmitarbeitern sowie Leistungs- und Qualitätseinbussen beim Produkt. Für die Projektbeteiligten auf operativer Ebene kann diese Situation zudem auch frustrierend sein, wenn z.B. das

Management – von strategischen Vorgaben abgesehen – wenig Engagement und Interesse für das Projekt auf operativer Ebene zeigt. Deshalb ist nach aufbauorganisatorischen Strukturen zu suchen, welche den Projektmitarbeitern erlauben, sich an solchen Entscheidungen zu beteiligen, und Spielräume offenlassen. Die Projektmitarbeiter sollten in diesem Sinne Einflussmöglichkeiten auf folgende Entscheidungen haben:

- Planung und Festlegung von Teilzielen,
- Planung und Festlegung des Projektmanagements auf operativer Ebene,
- Budgetierung und Terminierung von Teilaufgaben.

2. Eine repräsentative und aktive Mitarbeit der Benutzer ist nicht vorgesehen.

Eine organisatorische Integration der Benutzer in den Lenkungsausschuss, in Projektgruppen und/oder in Arbeitsgruppen als Voraussetzung für eine repräsentative und aktive Mitarbeit, bei der Benutzervertreter

- an den verschiedenen Projektphasen beteiligt sind,
- zu wesentlichen Planungs- und Gestaltungsfragen beigezogen werden,
- Systemvorschläge entwickeln und auch
- Mitentscheidungsmöglichkeiten haben,

ist nicht vorgesehen und auch nicht möglich. Mögliche Folgen davon sind Softwarelösungen, die an den Bedürfnissen der Benutzer vorbeizielen und zu Unzufriedenheit bei den Benutzern führen. Traditionelle Modelle der Aufbauorganisation bei der Softwareentwicklung sind daher u.a. vor dem Hintergrund einer unzureichenden Benutzerbeteiligung abzulehnen.

3. Die Projektorganisation ist funktionsorientiert und stark arbeitsteilig.

Traditionell werden vor allem grosse Projekte – die in entsprechend grossen Softwarefirmen und/oder grossen firmeninternen Entwicklungsabteilungen realisiert werden – funktional in Bereiche wie z.B. Systemanalyse, -konzept, Programmierung, Integration, Ausbildung usw. gegliedert. Bürokratie, lange Dienstwege sowie eine grosse Arbeitsteilung und Spezialisierung sind kennzeichnend für solche Strukturen. Ein entsprechender Aufwand für Administration und Koordination sowie eingeschränkte Flexibilität sind einige Folgen dieser Form der Projektorganisation.

In diesem Zusammenhang ist das aus dem soziotechnischen Systemkonzept abgeleitete Prinzip der *'Schaffung von relativ unabhängigen Organisations- bzw. Projekteinheiten',* in denen Schwankungen und Störungen selbst reguliert werden können, von besonderer Bedeutung. Dieses Prinzip impliziert, Projektgruppen nicht nach funktionalen Gesichtspunkten, sondern produkt- bzw. modulorientiert zu bilden. Dementsprechend sollten grosse Projekte in abgeschlossene Teilprojekte gegliedert werden, damit vollständige Aufgabenstellungen – die sowohl analytische, konzeptionelle, spezifizierende, realisierende und kontrollierende Teiltätigkeiten umfassen – verschiedenen Projektgruppen zur Bearbeitung übertragen werden können. Damit werden die Projektgruppen mit vollständigen und somit motivierenden Aufgaben beauftragt, die u.a. das selbständige Setzen von Zielen, die Wahrnehmung von Planungsfunktionen, die Auswahl von Arbeitsmitteln, Ausführungsfunktionen sowie Kontrollfunktionen mit Resultatfeedback umfassen (vgl. Volpert 1987).

2.3.2 Alternative Modelle der Aufbauorganisation

Alternative Modelle der Aufbauorganisation sind in traditionellen Projektmanagementmodellen meist nicht vorgesehen. Die dargestellten Problembereiche lassen allerdings Zweifel aufkommen, ob in traditionellen Projektstrukturen in effizienter Weise aufgaben- und benutzungsangemessene Software entstehen kann.

In diesem Abschnitt werden deshalb drei Organisationsmodelle beschrieben, die den dargestellten Problemen entgegenwirken und die Kooperation zwischen Softwareentwicklern und Benutzern unterstützen (siehe dazu auch Kubicek 1979).

In allen Modellen werden die Instanzen 'Lenkungsausschuss', 'Projektleitung' und 'Projektgruppe' unterschieden. Diese Instanzen haben generell die im folgenden angegebenen Aufgaben.

Lenkungsausschuss

Grundsätzlich übernimmt der Lenkungsausschuss als strategisches Organ die globale Steuerung, Überwachung und Kontrolle des Projektes. Eine interessenausgleichende Zielvorgabe, das Bereitstellen finanzieller, personeller und zeitlicher Ressourcen, das Setzen von Prioritäten sowie die Unterstützung des Projektleiters und der Projektgruppen auf operativer Ebene gehören zu diesem Aufgabenspektrum. Dies beinhaltet auch, dass der Lenkungsausschuss z.B. Aufträge für die Projektgruppe vorbereitet, Aufträge

freigibt, Anfragen der Projektgruppe beantwortet und zu deren Arbeitsergebnissen Stellung nimmt.

Projektleitung

Die Projektleitung hat für die projektzielgerechte Entwicklung des Systems zu sorgen. Daher steuert und koordiniert der Projektleiter die Aktivitäten zwischen Lenkungsausschuss, Projekt- und Arbeitsgruppen. Die Funktion der Projektleitung zielt – im Unterschied zur traditionellen Rolle der Projektleitung – u.a. darauf ab, Rahmenbedingungen zu schaffen, die eine motivierte und effiziente Projektarbeit auf operativer Ebene fördern (siehe dazu ausführlich Teil B, Kapitel 2.3.3).

Projektgruppe

Die Projektgruppe plant, realisiert und kontrolliert das Projekt auf operativer Ebene. Dies umfasst neben organisatorischen Aufgaben v.a. auch inhaltliche Aufgaben wie Systemanalyse, Anforderungsermittlung, Systemkonzeption, Systemspezifikation, Implementation, Systemtests, Systemeinführung sowie die begleitende Qualitätssicherung (siehe dazu bei Wallmüller 1990; siehe auch Teil B, Kapitel 2.3.4).

Die im folgenden beschriebenen Modelle zeigen unterschiedliche Möglichkeiten der Aufbauorganisation sowie Varianten der Zusammensetzung von Projektgremien. In Abhängigkeit von Projektzielen, dem Projekttyp und weiteren Rahmenbedingungen können geeignete Modelle für konkrete Vorhaben ausgewählt und bez. der konkreten Aufgaben und Kompetenzen der einzelnen Instanzen sowie der Informationsflüsse zwischen den Instanzen spezifiziert werden.

Da heutzutage meist verschiedene Fachbereiche in einem Unternehmen von der Entwicklung neuer Anwendungen oder der Weiterentwicklung bestehender Software betroffen sind, wird in den folgenden Modellen der *organisatorischen Integration von Fachbereichen* besondere Bedeutung beigemessen. So können z.B. bei der Entwicklung und Einführung eines integrierten Bürokommunikationssystems in einem Handelsunternehmen die aufgaben- und mitarbeiterspezifisch unterschiedlichen Interessen nur durch den Einbezug der Bereiche Verkauf, Kalkulation, Rechnungswesen, administrative Auftragsabwicklung, Einkauf, Lager und Versand in das Projekt angemessen berücksichtigt werden (zur repräsentativen Vertretung von Anwendern, Benutzern siehe ausführlich Teil A, Kapitel 2.3.1).

Das Modell der gemischten Projektgruppen

Beim Modell der gemischten Projektgruppen setzt sich die Projektgruppe aus internen und/oder externen Spezialisten wie z.B. Softwareentwicklern, Arbeitswissenschaftlern usw. und Vertretern der verschiedenen Fachbereiche zusammen (vgl. Abbildung 6).

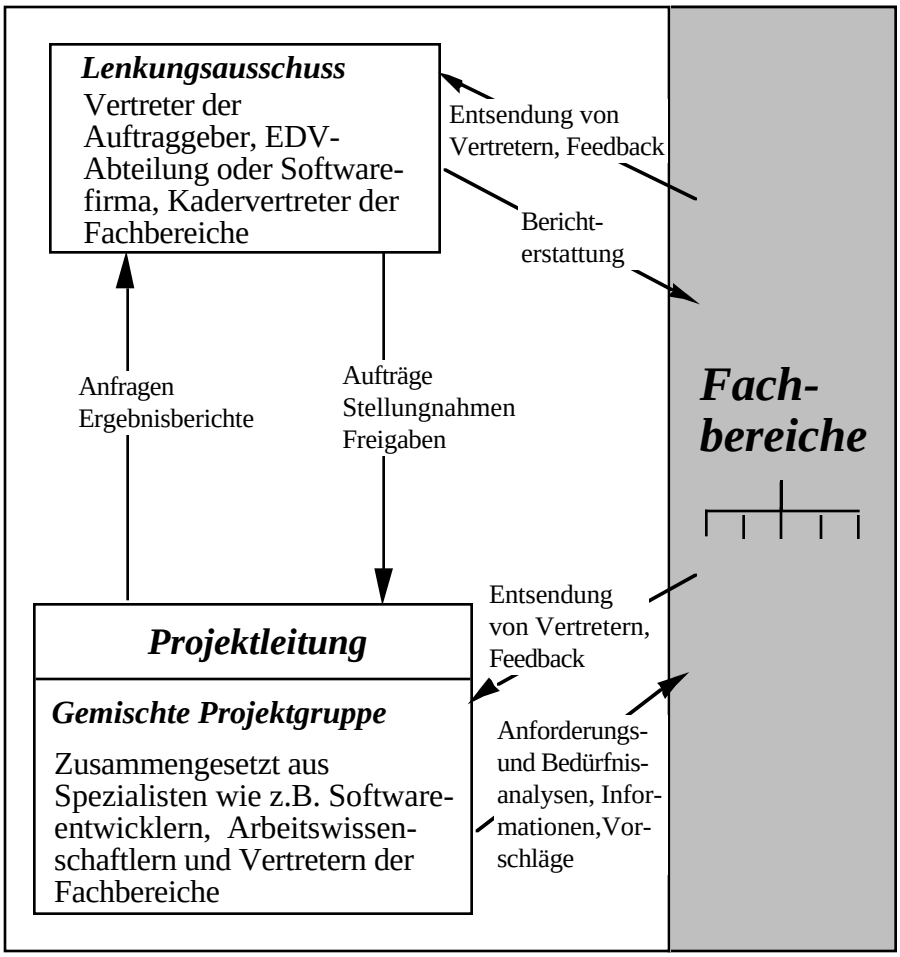


Abbildung 6: Das Modell der gemischten Projektgruppe.

Die Aufgabe der Projektgruppe ist die Konzeption und Realisierung des Systems. Die Anforderungen und Bedürfnisse der verschiedenen Fachbereiche werden im Rahmen von Anforderungs- und Bedürfnisanalysen durch die Projektgruppe erhoben. Die Fachbereiche erhalten von der Projektgruppe Informationen und Systemvorschläge. Die Fachbereiche wiederum geben – z.B. im Rahmen von Informationsveranstaltungen über das Projekt – der Projektgruppe Feedback (vgl. Abbildung 6).

Mögliche Aufgaben der Vertreter aus den Fachbereichen sind – gemeinsam mit den Softwareentwicklern und/oder Arbeitswissenschaftlern –, das bestehende technisch-organisatorische System zu analysieren, Anforderungen für das neue System bez. Funktionalität, Datenstrukturen usw. zu formulieren sowie die Entwicklung möglicher Systemlösungen, welche den Fachbereichen zur Beurteilung vorgeschlagen werden können. Um diese Aufgaben auch erfüllen zu können, müssen die Fachbereichsvertreter qualifiziert und von ihren täglichen Aufgaben zumindest zum Teil freigestellt werden (vgl. Teil A, Kapitel 2.3.1).

Die Arbeit der Projektgruppe unterliegt der Überwachung, Steuerung und Kontrolle des Lenkungsausschusses, in welchem z.B. Vertreter des Auftraggebers, Vertreter der internen EDV-Abteilung oder externen Softwarefirma sowie Kadervertreter der Fachbereiche wirken. Die Vertreter der Fachbereiche im Lenkungsausschuss bringen Anregungen, Anforderungen usw. ihres Fachbereiches im Lenkungsausschuss ein, wie sie auch umgekehrt die Fachbereiche über die Aktivitäten im Projekt informieren.

Der Vorteil dieses Modells besteht darin, dass Fachbereichsvertreter aktiv an der Planung und Realisierung des Projektes mitarbeiten, Anforderungen und Anregungen einbringen und dabei auch die verschiedenen Fachbereiche direkt über die laufenden Arbeiten auf strategischer und operativer Ebene informiert werden können. Bei Projekten mit langer Dauer besteht bei dieser Form des Benutzereinbezugs allerdings auch die Gefahr, dass die Benutzervertreter in der gemischten Projektgruppe im Laufe des Projektes den Kontakt zu ihrem Fachbereich verlieren und sich zu 'umgeschulten Systemspezialisten' entwickeln. Insofern bedingt diese Form der Projektorganisation, dass der Austausch zwischen Benutzervertretern und Fachbereichen z.B. im Rahmen von Informationsveranstaltungen, Workshops institutionalisiert wird bzw. kontinuierlich stattfindet.

Das Modell paralleler Arbeitsgruppen

Beim Modell paralleler Arbeitsgruppen wird ebenfalls von einer gemischten Projektgruppe ausgegangen. Darüber hinaus werden zusätzlich Arbeitsgruppen eingerichtet, die entweder interessen- oder aufgabenspezifisch zusammengesetzt sind.

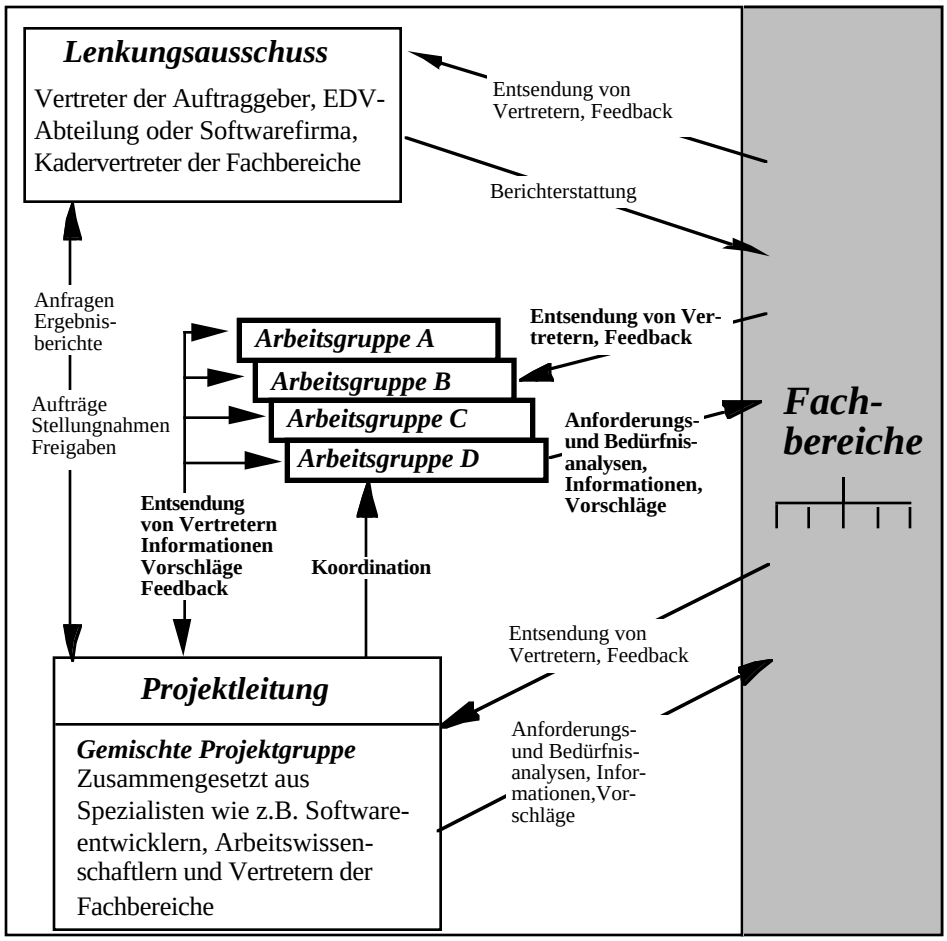


Abbildung 7: Das Modell paralleler Arbeitsgruppen.

Eine *interessenspezifische Arbeitsgruppe* setzt sich aus Vertretern verschiedener Fachbereiche – z.B. nur Abteilungsleitern – zusammen. Hierbei kann einem eventuell notwendigen Willensbildungs- und Abstimmungsprozess der einzelnen Fachbereiche auf taktischer Ebene Rechnung getragen werden.

Eine *aufgabenspezifische Arbeitsgruppe* setzt sich aus Vertretern eines oder mehrerer Fachbereiche zusammen und ist mit der Bearbeitung spezifischer Aufgaben wie z.B. der Entwicklung von möglichen Varianten der Ablauforganisation, des Bildschirm-Layouts, der Formulargestaltung usw. beauftragt (vgl. Abbildung 7).

Die einzelnen Arbeitsgruppen sollten die Grösse von ca. acht bis zehn Personen nicht übersteigen. Erfahrungsgemäss zeigt sich, dass in Arbeits- oder Projektgruppen mit dieser Grösse u.a. die Transparenz des 'sozialen Systems' erhalten bleibt und die Gefahr, dass verschiedene Subgruppen entstehen, reduziert wird. Mit dieser Grösse können gute Synergien entstehen, indem verschiedene Informationen und Wissensbestände genutzt werden.

Ein oder zwei Vertreter der Arbeitsgruppen wirken in der Projektgruppe mit und nehmen zum Austausch von Informationen, Vorschlägen und Feedback die Aufgabe von Verbindungsgliedern zwischen ihrer Arbeitsgruppe und der Projektgruppe wahr. Allgemeine Anforderungs- und Bedürfnisanalysen bei der Gesamtheit der Projektbetroffenen werden entweder durch die Projektgruppe oder durch die Arbeitsgruppen durchgeführt.

Dieses Modell ist vor allem für komplexe Projektvorhaben geeignet, bei denen umfangreiche Veränderungen des technisch-organisatorischen Systems in den Fachbereichen vorgesehen sind. Das Modell paralleler Arbeitsgruppen ermöglicht für solche Vorhaben eine 'modulare' Arbeit unter intensiver Mitarbeit der Fachbereiche. Dies setzt allerdings voraus, dass den Arbeitsgruppen in sich geschlossene Aufträge zur Bearbeitung übertragen werden.

Die adäquate Koordination der Arbeitsgruppen, welche bei diesem Modell durch die Projektleitung erfolgt, ist eine wichtige Voraussetzung für das Funktionieren dieser Form der Projektarbeit. Der Aufwand für diese Aufgabe ist entsprechend gross und ist in Projekten mit mehreren parallelen Arbeitsgruppen sinnvollerweise durch eine Assistenz des Projektleiters bzw. einen Projektkoordinator zu erfüllen.

Das Modell koordinierter Arbeitsgruppen

Eine weitgehend autonome Projektarbeit durch die Fachbereiche ist im Modell koordinierter Arbeitsgruppen vorgesehen. Dabei werden vor allem aufgabenspezifische Arbeitsgruppen eingerichtet, die sich aus Vertretern der Fachbereiche zusammensetzen. Der Projektleiter bzw. Projektkoordinator steuert und koordiniert die Arbeitsgruppen (vgl. Abbildung 8).

Hinter diesem Modell steht die Idee, dass in den Arbeitsgruppen die Analyse, Konzeptions- und Implementierungsaufgaben eines Projektes selbständig erfüllt werden bzw. die Verantwortung für deren Erfüllung bei den Arbeitsgruppen liegt. Programmiersprachen der vierten Generation bieten durchaus die Möglichkeit, dass Benutzer einfachere Applikationen selbständig realisieren. Für Aufgaben, die die Möglichkeiten der Arbeitsgruppen übersteigen, ist bei diesem Modell vorgesehen, dass Spezialisten (z.B. Softwareentwickler, Anwendungsspezialisten, Arbeitswissenschaftler) die Arbeitsgruppen beraten und in Abstimmung mit dem Lenkungsausschuss Aufträge der Arbeitsgruppen (z.B. Prototypenentwicklung, Programmierung) übernehmen.

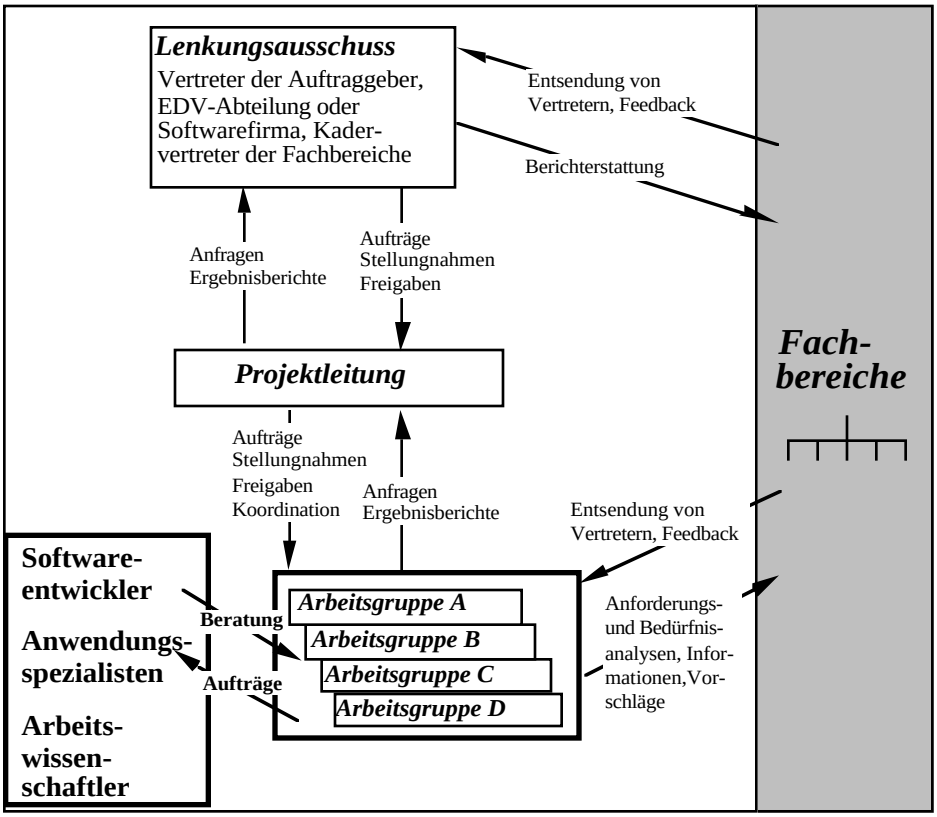


Abbildung 8: Das Modell koordinierter Arbeitsgruppen.

Voraussetzung für den Erfolg solcher koordinierter Arbeitsgruppen sind eine

(1) weitreichende fachliche und EDV-bezogene Qualifikation der Fachbereichsvertreter,

(2) ihre zeitliche Freistellung,

(3) eine modulare Auftragszuteilung in die Arbeitsgruppen sowie

(4) eine adäquate Steuerung und Koordination der Arbeitgruppen durch die Projektleitung.

Zusammenfassung

Die dargestellten Modelle unterscheiden sich deutlich in der organisatorischen Integration und den daraus resultierenden Möglichkeiten des Einflusses der Fachbereiche auf die Systementwicklung. Die Entscheidung für eines der Modelle und dessen projektspezifische Anpassung setzen die Identifikation der relevanten Interessengruppen, eine genaue Zieldefinition sowie eine Projektklassifikation auf der Basis der Rahmenbedingungen des Projektes voraus.

Bei Spezialprojekten wie z.B. der Entwicklung eines Expertensystems oder Projekten des Typs D empfiehlt sich ein Organisationsmodell, das sich weitgehend auf den Einsatz von Spezialisten stützt.

Bei 'einfacheren' Projektvorhaben des Typs A, B und C, die sich durch geringe Komplexität und Innovativität sowie einen homogenen Benutzer- und Aufgabenkontext auszeichnen, wäre z.B. das Modell 'Gemischte Projektgruppe' sinnvoll. Projekte mit hoher Komplexität und Innovativität sowie heterogenem Benutzerkreis und Aufgabenkontext erfordern dagegen die Realisierung einer auf parallelen oder koordinierten Arbeitsgruppen gestützten Aufbauorganisation.

Die erfolgreiche Praktizierung der beiden letzten Modelle setzt voraus, dass folgende Bedingungen erfüllt sind:

(1) Den Fachbereichsvertretern werden angemessene zeitliche Freiräume für die Projektarbeit gewährt.

(2) Die Fachbereichsvertreter verfügen über die notwendige EDV-Qualifikation.

(3) Die Fachbereichsvertreter besitzen grosses fachliches Anwendungswissen.

Zu den qualifikatorischen Voraussetzungen aktiver Mitarbeit von Fachbereichsvertretern wurde ausführlich im Teil A, Kapitel 2 eingegangen.

2.3.3 Die Projektleitung

Die zentrale Bedeutung des Projektleiters für eine erfolgreiche Abwicklung von Projekten wird immer wieder betont. Eine lehrbuchmässige Definition des Auftrages eines Projektleiters lautet wie folgt: "Der Projektleiter hat dafür zu sorgen, dass das spezifizierte Projektziel im gegebenen Kosten- und Terminrahmen erreicht wird." Diese Definition umfasst u.a. die Planung und Steuerung des Projektablaufes und Mitteleinsatzes, die Steuerung und

Koordination der Projektgruppe(n) auf operativer Ebene sowie die Ergebniskontrolle. Traditionell kommt dem Projektleiter dabei eine zentrale Rolle zu.

Im Unterschied zu dieser traditionellen Rolle wird dem Projektleiter gemäss den Prinzipien des soziotechnischen Systemkonzeptes eine anders akzentuierte Rolle zugewiesen. Der Projektleiter hat hierbei die Hauptaufgabe, die *Grenzbedingungen* im Projekt *zu regulieren*.

Wie bereits besprochen beinhaltet dies, dass die Aktivitäten des Projektleiters darauf abzielen, Bedingungen zu schaffen, die eine effiziente und motivierte Projektarbeit fördern. Zur Unterstützung der Arbeit der Projektgruppe nimmt der Projektleiter dabei eine 'Vermittler- und Moderatorenrolle' ein, die interne Regulation erfolgt jedoch durch die Projektgruppe selbst.

Im Zusammenhang mit dieser Moderatorenrolle bzw. diesem delegativen Führungsstil hat der Projektleiter folgende Aufgaben zu erfüllen:

(1) Zielvereinbarung zusammen mit der Projektgruppe,

(2) gemeinsame Überprüfung der Teilaufgaben nach Zielvereinbarung,

(3) Bereitstellung und Kontrolle der notwendigen Ressourcen,

(4) Bereitstellung der relevanten Informationen für die Projektarbeit,

(5) Bereitstellen von Qualifizierungsmöglichkeiten für die Projektarbeit,

(6) gemeinsame Evaluation der Ergebnisse.

Diese Rollenzuschreibung hat den Vorteil, dass der Projektleiter von operativen Aufgaben entlastet ist und die notwendigen Rahmenbedingungen für eine effiziente Aufgabenausführung in einer selbstregulierenden Projektgruppe (vgl. Teil B, Kapitel 2.3.4) schaffen kann.

Kasten 17: Ein Beispiel für die grenzregulierende Funktion des Projektleiters.

Der Projektleiter bespricht im Auftrag des Lenkungsausschusses mit der gemischten Projektgruppe, die sich aus drei Analytikern, Programmierern sowie drei Benutzern der Fachabteilung zusammensetzt, die Ziele für den nächsten Projektabschnitt. Man einigt sich darauf, die bestehenden Aufgaben und Arbeitsabläufe, die Informationsflüsse und Datenstrukturen sowie die technischen Ausführungsbedingungen in der Fachabteilung zu analysieren. Die Analyse soll in 16 Kalenderwochen durchgeführt und ergebnisorientiert dokumentiert werden. Das methodische Vorgehen bleibt – unter Berücksichtigung firmeninterner Standards – weitgehend der Projektgruppe überlassen.
Dieses Aufgabenpaket und der Zeitplan werden vom Projektleiter dem Lenkungsausschuss vorgelegt und von diesem genehmigt. Der Projektleiter informiert daraufhin die Fachabteilung über das Vorhaben, nimmt Anregungen aus der Fachabteilung auf und meldet diese der Projektgruppe zurück. Die Arbeiten werden nach internen Qualifizierungsmassnahmen von der Projektgruppe aufgenommen.
Der Projektleiter wird im Abstand von 4 bis 5 Wochen über den Stand der Arbeiten informiert, Probleme werden thematisiert, der Projektleiter gibt Anregungen für die weitere Arbeit. Nach Ablauf der 16. Kalenderwoche setzen sich die Projektgruppe und der Projektleiter wiederum zusammen, um den Stand der Analysen, deren Ergebnisse sowie das weitere Vorgehen zu besprechen. Der Lenkungsausschuss wird im Anschluss daran durch den Projektleiter informiert.

Die grenzregulierende Funktion des Projektleiters umfasst weitere Aufgaben wie die Koordination verschiedener Projekt- oder Arbeitsgruppen auf operativer Ebene. Die Schnittstelle zwischen strategischer und operativer Ebene im Projekt ist ebenfalls im Projektleiter personifiziert. In dieser Rolle kommt dem Projektleiter auch eine politische Funktion zu. Er hat das Projekt 'nach aussen' zu vertreten, für das Vorgehen und die Lösungsansätze zu werben bzw. diese – falls notwendig – zu verteidigen sowie zu konsensorientierten Entscheidungen und Lösungswegen im Projekt zu verhelfen. Die Aufgaben des Projektleiters sind in sozialer wie fachlicher Hinsicht anspruchsvoll; die für die erfolgreiche Aufgabenerfüllung notwendigen Qualifikationen sind vielschichtig. Eine angemessene Ausübung dieser Funktion setzt voraus, dass er mit entsprechenden Entscheidungskompetenzen

ausgestattet ist. Die Wahl des Projektleiters erfordert besondere Sorgfalt, das Bereitstellen von Qualifizierungsmöglicheiten sowie ein Verständnis für diese Rolle, das Fehler zulässt bzw. auf Lern- und Entwicklungsfähigkeit aller Beteiligten setzt.

Folgende Qualifikationen und Bedingungen sind für die Arbeit des Projektleiters nach dem skizzierten Verständnis wichtige Voraussetzungen (vgl. Bartsch-Spörl 1991).

Fachkompetenz: Fachbereich- und EDV-Wissen

Der Projektleiter benötigt sowohl ausreichendes Wissen über den Fachbereich, das heisst, dessen Aufgaben und technisch-organisatorische Ausführungsbedingungen als auch fundiertes Wissen über Möglichkeiten und Grenzen der EDV-Technik.

Methodenkompetenz

Der Projektleiter braucht ebenso fundiertes Wissen über Anwendungsmöglichkeiten und Nutzen sowie Grenzen verschiedener Methoden zum Projektmanagement und zur Softwareentwicklung, insbesondere auch über Methoden zur Analyse, Darstellung, Produktion und Evaluation.

Sozialkompetenz

Der Projektleiter benötigt viel Wissen und Erfahrung im Umgang mit Menschen; dies umfasst u.a. Wissen und Erfahrung über Gruppendynamik, Kommunikation und Führungsverhalten. Da Softwareprojekte sehr unterschiedliche Interessengruppen berühren, sind soziale Konflikte oft 'vorprogrammiert'. Es ist die Aufgabe des Projektleiters, mit derartigen Konflikten konstruktiv umzugehen, indem er interessenausgleichende Lösungswege aufzeigt bzw. herbeiführt.

Positives Image und Unterstützung

Der Projektleiter sollte bei den Projektbeteiligten über ein positives Image verfügen, damit er durch die verschiedenen Interessengruppen in seiner Arbeit unterstützt wird und auch seine Funktion als Verbindungsglied zwischen den Interessengruppen ausüben kann.

Identifikation und Empathie

Der Projektleiter muss sich mit den Zielen und Inhalten des Projektes identifizieren, damit er das Projekt wirkungsvoll und glaubwürdig als Promotor vertreten kann.

Unter Empathie ist Einfühlungsvermögen und eine offene Wahrnehmung zu verstehen für das, was im Projekt geschehen kann, real geschieht oder auch nicht geschieht. Mögliche Gefahren sollten dem Projektleiter bekannt sein und entsprechende Warnsignale von ihm frühzeitig, richtig und vollständig erkannt und gedeutet werden (siehe Übersicht 1 im Anhang).

Aus verschiedenen Erfahrungsberichten von Softwareprojekten sind vor allem folgende Gefahren und Warnsignale bekannt:

(1) Man überschätzt sich mit dem Projekt bzw. nimmt sich zuviel vor.

(2) Die Projektziele sind unklar oder schliessen sich gegenseitig aus.

(3) Dem Projekt fehlt ein (Macht-)Promotor.

(4) Die Komplexität der fachtechnischen Aufgabenstellung wird unterschätzt.

(5) Das Projekt ist 'angeordnet' und vom Fachbereich nicht gewollt.

(6) Das Projekt wird kosten- und terminbezogen viel zu knapp budgetiert.

(7) Die organisatorischen Gegebenheiten im Fachbereich werden vor der Systemkonzeption nicht hinterfragt.

(8) Es fehlen angemessene Methoden und Werkzeuge.

(9) Der Zeitplan 'gerät schon sehr früh ins Wanken'.

(10) Die Qualifikationen der Mitglieder im Team sind unzureichend.

(11) Die Ergebnisse werden mangelhaft dokumentiert.

(12) Der Softwareentwurf lässt keine nachträglichen Anforderungsänderungen zu.

(13) In der Projektorganisation fehlen die Benutzervertreter.

(14) Die Qualitätssicherung wird vernachlässigt.

(15) Der Projektgruppe fehlt es an Motivation.

Im folgenden Abschnitt werden vorwiegend Gestaltungshinweise für die Arbeit in der Projektgruppe gegeben.

2.3.4 Die Projektgruppe

In vielen modernen Projektmanagementbüchern wird übereinstimmend Team- bzw. Gruppenarbeit als die adäquate Arbeitsorganisation für die Abwicklung von Projekten dargestellt. Die Ergebnisse einer Meta-Analyse zu Projekterfolgskriterien (Gemünden 1990) zeigen, dass sehr verschiedene Merkmale einer Projektgruppe wie z.B. Fachkompetenz, Entscheidungskompetenzen, Partizipation, Motivation, Kontinuität, Fluktuation im Zusammenhang mit Projekterfolgskriterien stehen. Das Verständnis darüber, wie Teamarbeit zu gestalten ist, damit eine kompetente und motivierte Aufgabenerfüllung erfolgt, ist allerdings in der betrieblichen Praxis sehr unterschiedlich.

In Abschnitt 2.3.1 von Teil B wurde bereits darauf hingewiesen, dass in traditionellen Organisationsmodellen die Arbeit in Projektgruppen häufig funktional und sehr arbeitsteilig gestaltet ist. In grossen Projekten werden z.T. Teams für Analyse und Konzeptentwicklung, andere für die Spezifikation und Realisierung und wiederum andere für die Qualitätssicherung eingerichtet. In kleineren Projekten dagegen wird zwar häufig nur eine Projektgruppe eingesetzt, die die gesamten Aufgaben des Softwarelebenszyklus bearbeitet; solche Gruppen sind jedoch häufig intern funktionsorientiert und arbeitsteilig ausgerichtet und zusammengesetzt (vgl. Strohm & Ulich 1991). Programmierer werden vorwiegend mit programmiertechnischen und testenden Aufgaben beauftragt, dagegen selten mit analytischen und konzeptionellen Aufgaben, die primär zum Aufgabenbereich des Projektleiters und Systemanalytikers gehören.

Programmierer, die unter solch arbeitsteiligen Bedingungen arbeiten, sind mit ihrer Arbeitssituation häufig sehr unzufrieden, da für sie Kooperationsmöglichkeiten weder mit anderen Programmierern noch mit Benutzern speziell in der Analyse- und Konzeptionsphase bestehen. Durch mangelnden Überblick über eine komplexe Aufgabe erleben sie ihre Arbeit als partialisiert und entfremdet (vgl. Hacker 1989). Dies ist gut nachvollziehbar, wenn man bedenkt, dass Softwareentwickler selbst vor allem die analytischen und konzeptionellen Aufgaben – im Unterschied zu den realisierenden Aufgaben – als fachlich und sozial herausfordernd und somit qualifizierend erleben (vgl. Frese et al. 1991)!

Aus starker Funktions- und Arbeitsteilung in und zwischen Projektgruppen resultieren Mangel an Transparenz, Eigeninitiative und Verantwortungsübernahme bei den Gruppenmitgliedern, die Lern- und Entwicklungsmöglichkeiten bei der Arbeit sind eingeschränkt, geringe Motivation und Pro-

duktivität können die Folge davon sein. Dies ist in der betrieblichen Praxis offensichtlich erkannt worden, was sich u.a. daran zeigt, dass verstärkt Analytiker/ Programmierer – die mit der Bearbeitung möglichst vieler Aufgaben des Softwareentwicklungsprozesses beauftragt werden können – ausgebildet bzw. auf dem Arbeitsmarkt gesucht werden.

Effiziente, weitgehend selbständig arbeitende Projektgruppen sind (in Anlehnung an Ulich 1994) durch folgende Merkmale gekennzeichnet:

(1) Relative Unabhängigkeit der Projektgruppe,

(2) innerer Aufgabenzusammenhang in der Projektgruppe sowie

(3) Einheit von Produkt und Projektgruppe.

Dies bedeutet, dass einer Gruppe von Analytiker/Programmierern eine vollständige Projektaufgabe – welche idealtypischerweise möglichst viele interdependente Teilaufgaben von A bis Z umfasst – übertragen wird. In gemischten Gruppen wird eine solche Gruppe z.B. durch Benutzervertreter ergänzt. Die Übertragung einer vollständigen Projektaufgabe ist eine wesentliche Voraussetzung für das Verständnis einer gemeinsamen Aufgabe in der Gruppe. Dieses Verständnis ermöglicht ein höheres Mass an Selbstregulation und gegenseitiger Unterstützung (vgl. Ulich 1994). Selbstregulation beinhaltet dabei, dass die Gruppe neben inhaltlichen Aufgaben auch organisatorische Aufgaben wahrnimmt. Die Organisation der Arbeit erfolgt weitgehend selbständig durch die Projektgruppe.

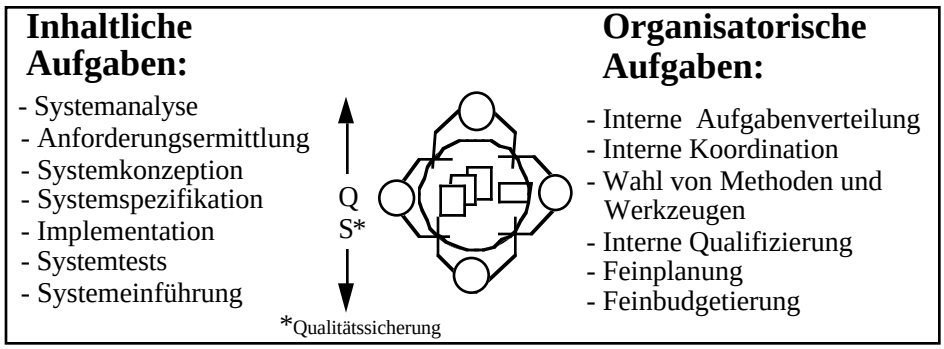


Abbildung 9: Inhaltliche und organisatorische Aufgaben in einer selbstregulierenden Projektgruppe.

In der Abbildung 9 sind inhaltliche und mögliche organisatorische Aufgaben in einer selbstregulierenden Projektgruppe aufgeführt. Aus dieser Abbildung ist ersichtlich, dass in einer selbstregulierenden Projektgruppe neben der ganzheitlichen Projektaufgabe auch organisatorische Aufgaben wie die interne Aufgabenverteilung und Koordination, Feinplanung usw. wahrgenommen und ausgeführt werden. Dies beinhaltet, dass z.B. die Form der internen Koordination durch die Gruppe selbst festgelegt wird.

Im Sinne der arbeitswissenschaftlichen Forderung für die industrielle Produktion, Qualität nicht zu kontrollieren, sondern zu produzieren, ist die *begleitende Qualitätssicherung* über alle Projektschritte des Softwarelebenszyklus eine weitere Aufgabe der Projektgruppe. Dies beinhaltet, dass die Gruppe z.B. auch Reviews zu den Anforderungen und zum Entwurf durchführt. Eine wichtige Voraussetzung für die dargestellte Arbeitsform ist die soziale Integration. Deshalb sollte die Projektgruppe – für die Zeit der Projektarbeit – in einem gemeinsamen Büro oder zumindest räumlich nahe zusammen arbeiten, damit Kommunikation und Kooperation erleichtert werden (siehe auch DeMarco und Lister 1991).

Auf die Rolle des Projektleiters in einer selbstregulierenden Projektgruppe wurde in Kapitel 2.3.3 des Teils B bereits eingegangen. In grösseren Projekten, die den Einsatz mehrerer solcher Gruppen erfordern, ist eine wesentliche Funktion des Projektleiters die Koordination der verschiedenen Gruppen, damit konsistente Teilergebnisse entstehen, die zu einem Ganzen zusammengeführt werden können.

Die Aufteilung von grossen Projekten in kleine, überschaubare und ganzheitliche Teilprojekte im Sinne einer produktorientierten Arbeitsteilung – welche auf der Basis einer modularen Systemkonzeption erfolgen kann – bekommt in diesem Zusammenhang eine zentrale Bedeutung.

In einem sehr grossen Projekt, welches die Einrichtung von z.B. sechs selbstregulierenden Projektgruppen bedingt, ist die Schaffung einer Projektstruktur mit *überlappenden Gruppen* eine Notwendigkeit, damit verteiltes Wissen und verschiedene Arbeitsergebnisse integriert werden können. Die Entwicklung eines Gesamtkonzeptes des Systems oder eines übergreifenden Konzeptes zur Qualitätssicherung wäre zudem ein Beispiel für eine Projektaufgabe, welche in übergeordneten Gruppen ausgeführt werden kann bzw. muss.

Aus Abbildung 10 ist ersichtlich, dass übergeordnete Gruppen aus Vertretern der verschiedenen Projektgruppen gebildet werden. Die Vertreter in

diesen übergeordneten Gruppen werden entweder durch die Gruppe fest bestimmt oder wechseln nach einem Rotationsprinzip. Die Aufgaben dieser übergeordneten Gruppen können Koordination, Wissens- und Ergebnistransfer oder spezifische Aufträge umfassen.

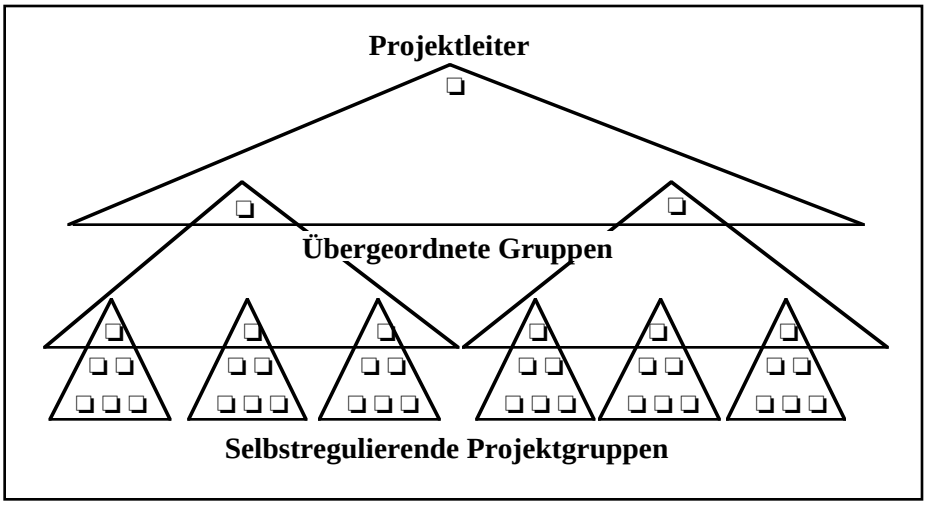


Abbildung 10: Ein Modell überlappender Projektgruppen (in Anlehnung an Likert 1972).

Vertreter mehrerer übergeordneter Gruppen bilden – falls notwendig – mit dem Projektleiter eine weitere Instanz, in der projektübergreifend Absprachen und Koordination stattfinden können. Eine Projektorganisation mit selbstregulierenden Gruppen kann folgende Vorteile zeigen:

(1) Sie fördert die Identifikation der Gruppe mit der Projektaufgabe.

(2) Die Arbeit in derartigen Gruppen ist qualifizierend in fachlicher wie in sozialer Hinsicht.

(3) Die Flexibilität, Kreativität und Motivation in der Projektgruppe wird dadurch gefördert.

(4) Diese Form der Projektorganisation wirkt der Bürokratisierung von Projekten entgegen.

Kontinuität in der personellen Zusammensetzung von Projekten, bei der Qualifikationen und Erfahrungen erhalten bleiben und nicht z.B. infolge von Fluktuationen verlorengehen, wird durch selbstregulierende Projektgruppen ebenfalls wahrscheinlicher bzw. besser erreicht. Damit sich diese Vorteile einstellen, sind jedoch einige Bedingungen zu erfüllen:

(1) Die Projektaufgabe sollte für die Gruppenmitglieder überschaubar sein.

(2) Die einzelnen Teilaufgaben sollten einen inneren Zusammenhang aufweisen.

(3) Die Projektgruppe sollte über vereinbarte Output-Ziele verfügen.

(4) Die Projektgruppe sollte über den Grad der Zielerreichung laufend Feedback erhalten.

Erfahrungsgemäss zeigt sich, dass Arbeits- bzw. Projektgruppen eine Grösse von acht bis zehn Personen nicht übersteigen sollten, damit neben der geforderten aufgabenbezogenen Transparenz auch die Transparenz des 'sozialen Systems' erhalten bleibt. Ergebnisfeedback schon während der Entwicklung ist wiederum am besten durch den Einsatz von Prototyping und eine enge Zusammenarbeit mit den Benutzern zu erreichen.

Eine weitere wichtige Voraussetzung für die Arbeit in selbstregulierenden Projektgruppen ist ein Entlohnungskonzept, das weniger auf einer individuellen Leistungsbemessung basiert als vielmehr die gesamte Gruppenleistung sowie die Qualifikation der einzelnen Gruppenmitglieder – das heisst ihr Können ('pay for knowledge') und damit ihre Flexibilität und Einsetzbarkeit – honoriert. Gruppenleistungsanteile, die z.B. für das Erreichen vereinbarter Qualitäts- oder Budgetziele vorgesehen sind, sollten gleichmässig auf alle Gruppenmitglieder verteilt werden. Dies ist eine weitere Möglichkeit, um die kollegiale Unterstützung in der Projektgruppe zu fördern.

2.4 FAZIT

Im Rahmen dieses Kapitels wurden u.a. die zentralen Elemente des Projektmanagements dargestellt sowie eine soziotechnische Betrachtung der Softwareentwicklung angestellt. Dabei kommt den Aufgaben der Zieldefinition und Projektklassifikation eine zentrale Bedeutung zu. Vor dem Hintergrund der Merkmale und Probleme traditioneller Formen der Aufbauorganisation bei der Softwareentwicklung wurden alternative Modelle vorgestellt. Das Hauptaugenmerk wurde dabei auf projektorganisatorische Mög-

lichkeiten der Fachbereichsvertretung bzw. des Benutzereinbezugs sowie eine Beschreibung der Aufgaben des Projektleiters und der Projektgruppe gelegt. Im Vergleich zu traditionellen Projektmanagementmodellen basieren die beschriebenen Ansätze auf dezentralen Strukturen, Benutzerorientierung und Selbstregulation. Vielfältige Erfahrungsberichte bestätigen den Erfolg dieser Formen der Arbeits- und Organisationsgestaltung (siehe insbesondere DeMarco und Lister 1991).

Dennoch stossen solche Ansätze immer wieder auf Widerstand. Ein Grund dafür liegt sicherlich darin, dass diese Konzepte etablierte Machtstrukturen bedrohen. Ulich (1991) kommt in diesem Zusammenhang zu folgendem Fazit: "Das Problem, um das es sich hier handelt, ist offenbar mindestens so alt wie unsere Zeitrechnung. Es wurde bereits in einer Schilderung benannt, die Petronius, dem römischen Satiriker und Senator zur Zeit Neros, zugeschrieben wird: 'Wir trainierten hart ... aber es schien, dass wir immer dann reorganisiert wurden, wenn wir gerade dabei waren, ein Team zu werden'."

Die Wahl zwischen traditionellen oder alternativen Organisationsmodellen zur Softwareentwicklung ist daher auch unmittelbar mit der folgenden Frage verknüpft: Entweder dient das Projekt der Erhaltung und Zementierung bestehender Arbeits-, Organisations- und Machtstrukturen – mit den entsprechenden Problemen – oder es kann dazu eingesetzt werden, die Entwicklung effizienter, aufgaben- und benutzungsorientierter Software zur Unterstützung innovativer Arbeitsabläufe umzusetzen.

Der Übergang von klassischen, funktional arbeitsteiligen hin zu zeitgemässen, funktional integrierten Arbeits- und Organisationsstrukturen kann aus arbeitspsychologischer Perspektive allerdings nur durch eine Form der Projektarbeit erreicht werden, welche u.a. die Kreativität und Innovativität der Projektmitarbeiter fördert. Die beschriebenen Ansätze können dazu einen Beitrag leisten.

3 Prozessgestaltung

In diesem Kapitel wird eine Bestandsaufnahme der bisher bekannten Problembereiche des traditionellen Phasenmodells (das 'Wasserfall'-Modell) vorgenommen, deren Überwindung zu einem iterativ-zyklischen Ablaufmodell führt. Dieses iterativ-zyklische Ablaufmodell (Quadrantenmodell) für partizipative Softwareentwicklungen macht aufgrund seiner zyklischen Struktur neue Vorgehensweisen notwendig, welche mit modernen Konzepten der Qualitätssicherung bei der Softwareentwicklung übereinstimmen.

3.1 Warum das Phasenmodell nicht genügt

Das traditionelle, lineare Phasenmodell der Softwareentwicklung (Frühauf, Ludewig und Sandmayr 1991a) unterscheidet die folgenden Phasen:

(1) Anforderungsanalyse,

(2) Aufgaben- und Systemanalyse,

(3) Spezifikation der Anforderungen,

(4) Systementwurf ggf. Modulspezifikation,

(5) Codierung ggf. Modultest,

(6) Integration und Systemtest,

(7) Installation und Abnahmetest,

(8) Betrieb und Wartung.

In diesem linearen Ablaufmodell ist es bei der Entdeckung von Fehlern, Problemen usw. nur möglich, zu der jeweils unmittelbar vorhergehenden Phase zurückzukehren. Oftmals werden jedoch auch Fehler aus früheren Phasen entdeckt. Die bekanntesten Schwächen des traditionellen Phasenmodells sind in Kasten 18 beschrieben.

Die Attraktivität des Phasenmodells kommt daher, dass sich die einzelnen Phasen scheinbar logisch direkt auf der linearen Zeitachse abbilden lassen. Somit wird das Phasenmodell aber als Projektmanagementmethode zur Steuerung des Entwicklungsprozesses missverstanden. Dabei dienen die Phasenergebnisse als 'Meilensteine', ein beliebtes – wenn auch häufig unzureichendes – Controlling-Instrument. Für eine angemessene Ablauforganisation wird jedoch ein der inhaltlichen Struktur des Softwareentwick-

lungsprozesses angepasstes, von der Zeitdimension zunächst unabhängiges Modell benötigt. Erst in einem zweiten Schritt wird die zyklische Struktur hinsichtlich der notwendigen, unterschiedlichen Aktivitäten der unternehmensspezifischen Projektkultur auf der linearen Zeitachse abgebildet (siehe auch AMI 1992). Um diese Abstraktion zu begründen, werden Konzepte aus der Systemtheorie benötigt.

Kasten 18: Die bekanntesten Schwächen des traditionellen Phasenmodells (aus Hesse, Merbeth und Frölich 1992, S. 75).

"Häufig keine systematische Benennung und Trennung von Tätigkeiten, Ergebnissen und Phasen; keine über zwei benachbarte Phasen hinausgehenden Entwicklungszyklen; keine Querbezüge zwischen frühen und späten Entwicklungsphasen; lange Entwicklungsdauer zwischen Vorgaben (Spezifikation) und Realisierung des Systems bzw. seiner Bausteine; keine oder ungenügende Beachtung moderner Werkzeuge mit weitreichenden Generierungsmöglichkeiten, die den gesamten Projektablauf verändern; keine oder unzureichende Behandlung des Wartungsproblems; keine Vorkehrungen für die Kooperation zwischen Anwendern und Entwicklern, die Einbeziehung der Benutzer in den Entwicklungsprozess und evolutionäre Systementwicklung."

In der Systemtheorie wird zwischen den beiden Lenkungsprinzipien der 'Steuerung' und der 'Regelung' unterschieden. Das traditionelle Phasenmodell beruht primär auf dem Prinzip der 'Steuerung': Arbeitsaufträge werden von einer Phase an die nächste vergeben. Dabei wird davon ausgegangen, dass alle wesentlichen Voraussetzungen für die Auftragsbearbeitung in der folgenden Phase bereitstehen und es keine ungeplanten bzw. unvorhergesehenen Einflüsse gibt (z.B. mangelhafte Hardware, fehlende Ressourcen, unzureichende Qualifikation, Aktivitäten höherer Priorität usw.). Die Projektleitung hat ein komplettes Wissen über alle geplanten Folgeaktivitäten; es gibt drei Voraussetzungen für dieses 'komplette Wissen':

(1) Die genaue Kenntnis der Reaktionen aller technischen und sozialen Komponenten,

(2) die genaue Kenntnis aller, die geplanten Aktivitäten beeinträchtigenden Grössen ('Störgrössen') sowie

(3) ein hinreichendes Wissen über die notwendigen Interventionen, um allen Störgrössen adäquat und *im vorhinein* begegnen zu können.

Diese drei Voraussetzungen sind – selbst bei sehr erfahrenen Projektleitern – praktisch nie vollständig erfüllt (Schiemenz 1979, S. 1024).

Die Verwendung des sehr leistungsfähigen Lenkungsprinzips der 'Regelung' dagegen setzt im Grunde nur die Kenntnis der Aufträge voraus, die sich auf das Ergebnis einer Projektaktivität der Tendenz nach in die gewünschte Richtung auswirken. Übertragen wir das allgemeine Schema für Regelung auf den Kontext der Softwareentwicklung, so gelten als Optimalitätskriterien alle relevanten technischen und sozialen Faktoren. Der Soll-Ist-Wertabgleich bei der 'Regelung' überprüft, inwieweit die von aussen vorgegebenen Soll-Werte unter Einhaltung der Randbedingungen erfüllt sind. Dies kann durch die Grenzregulationsaufgaben der Projektleitung bewerkstelligt werden.

Natürlich kommt auch das Lenkungsprinzip der 'Steuerung' in aktuellen Softwareentwicklungen zur Anwendung. Hierbei ist an direktive Entscheidungen und Vorgaben durch den Auftraggeber, die Projektleitung oder andere Führungsgremien usw. zu denken. Oft arbeiten Steuerungssysteme wirtschaftlicher als entsprechende Regelungssysteme – *aber nur*, wenn die oben genannten Voraussetzungen zutreffen! Dies ist auch der Grund dafür, warum grundsätzlich in erster Linie versucht wird, ein gesteuertes System herzustellen, sprich: dem Phasenmodell so nahe wie möglich zu kommen. Nehmen wir jedoch die empirisch ausreichend belegte Tatsache *ernst*, dass niemand im vorhinein (z.B. vor Projektbeginn) ein hinreichendes Wissen über den zu 'steuernden' Prozess (z.B. komplexe, innovative Projekte) haben kann, so gilt es zu entscheiden, an welchen Stellen im Softwareentwicklungsprozess Optimierungszyklen *unabdingbar* sind.

3.2 DER OPTIMIERUNGSZYKLUS

Die zyklische Struktur des Regelkreises lässt sich nun mit einigen Anpassungen zu einem *Optimierungszyklus* im Rahmen der Softwareentwicklung ausbauen. Die Grundidee besteht darin, dass alle *Aktivitäten* durch eine entsprechende *Testkomponente* ergänzt werden (AMI 1992). Dies klingt zunächst aufwendiger, als es tatsächlich ist. Viele 'Tests' werden bereits

heute schon angewendet, ohne dass man sich dessen oftmals bewusst ist. Wir werden weiter hinten sehen, welche 'Tests' dies im Einzelnen sind.

Als 'Aktivität' können sehr unterschiedliche Verfahren, Methoden und Techniken in Frage kommen; dies richtet sich nach der Art des Ergebnisses. Während die 'Randbedingungen', welche von aussen auf einen Optimierungszyklus einwirken, bewusst und zielorientiert vorgegeben werden (z.B. Budget, Projektdauer usw.), sind die 'Störgrössen' unbeabsichtigt und unvorhersehbar. Wir bezeichnen den 'Test-Aktivitäts-Zyklus' als *Optimierungszyklus* (siehe Abbildung 11). Eine wesentliche Beschreibungsgrösse des Optimierungszyklus ist die benötigte Zeit für einen Durchlauf. Je grösser die Durchlauf-, das heisst, Zykluszeit ist, desto aufwendiger ist der jeweilige Optimierungszyklus. Ziel benutzerorientierter Softwareentwicklung ist es nun, möglichst effiziente Optimierungszyklen in den Entwicklungsprozess einzubauen.

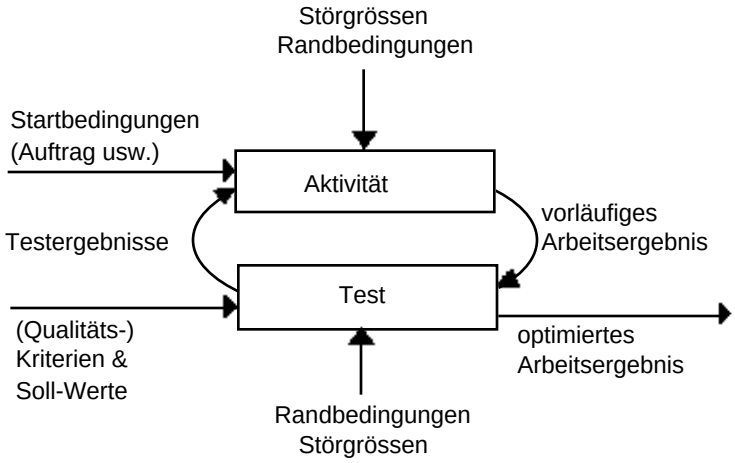


Abbildung 11: Die einzelnen Komponenten des Optimierungszyklus in Anlehnung an Deming (1989).

In den folgenden Abschnitten dieses Kapitels werden die einzelnen Abschnitte und Phasen von benutzerorientierter Softwareentwicklung vorgestellt und die in den einzelnen Phasen möglichen Aktivitäten zur Beteiligung von Benutzern diskutiert. Eine benutzerorientierte Softwareentwicklung

lässt sich am besten im Rahmen eines zyklischen Versionenkonzeptes realisieren. Wir teilen den gesamten Versionenzyklus in vier Quadranten auf:

Quadrant-I umfasst die Analyse und Bewertung des Arbeitssystems, in dem die Software eingesetzt werden soll; aus der Bewertung werden dann die Anforderungen abgeleitet. Durch die mehrfache Optimierung im *Quadrant-I* entsteht aus dem 'Grobkonzept' somit das 'Feinkonzept'.

Quadrant-II umfasst die Spezifikation (z.B. das Datenmodell) und den Entwurf, welcher dann im

Quadrant-III als Programm realisiert wird. Nach einer ersten Testphase der einzelnen Programmteile erfolgt die Integration; die auslieferbare Version wird dann im

Quadrant-IV im Arbeitssystem eingesetzt und benutzt.

3.3 QUADRANT-I: DIE ANALYSE- UND BEWERTUNGSPHASE

Als erstes muss bestimmt werden, 'ob' und 'wo' ein sinnvoller Einsatz von Technologie möglich bzw. erforderlich ist. "Die Vorstellung, man könne mit Hilfe der Technik die Defizite einer Organisation beheben, ohne die Strukturen der Gesamtorganisation in Frage zu stellen, ist zwar noch immer weitverbreitet, aber zumeist ein Trugschluss" (Klotz 1991, S. 108). Wichtig ist, das Arbeitssystem als eine *lebendige Organisation*, als einen sich selbst erhaltenden Organismus zu begreifen, der sich entwickeln und verändern muss, um die Organisationsziele zu erreichen. Unter dieser Perspektive geht es bei der Festlegung der Organisationsschnittstelle primär darum, die Lebensfähigkeit der Organisation durch Einsatz von Technologie zu fördern bzw. zu erhalten. Hierzu wird ein fundiertes Wissen für arbeits- und organisationsgestalterische Massnahmen benötigt (siehe Übersicht 3 im Anhang).

Alle Bemühungen – seitens der Softwaretechnik – zur *Prozessmodellierung* können nicht an der Tatsache vorbeigehen, dass die wesentlichen Aktivitäten in dieser Phase *Prozessmoderationen* sind. Gerade das Agieren im organisatorischen Umfeld ist von entscheidender Bedeutung. Erst eine geeignete Prozessmoderation erlaubt es dem Unternehmen, selbst herauszufinden, welche Veränderungen wünschenswert, notwendig und möglich sind.

Da die meisten Softwareentwicklungsprojekte primär von Softwaretechnikern durchgeführt werden, diesen aber oft eine ausreichende Ausbildung in

Prozessmoderation fehlt, ist die Analysephase oft die am meisten vernachlässigte Phase. In der angewandten Psychologie wurden und werden primär kommunikative Methoden und Techniken zur Prozessmoderation entwickelt, welche dann in der Arbeits- und Organisationspsychologie übernommen, angepasst und angewendet werden. Der Nutzen dieser Methoden wird von Softwareentwicklern nach eigenen Angaben häufig unterschätzt.

Die Fehlerbehebungskosten in späteren Phasen, die durch eine suboptimale Analyse entstehen, sind im Vergleich zu einem grösseren Analyseaufwand unverhältnismässig hoch: Faktor 100 bis 1000! (siehe dazu Boehm 1981). Es ist an der Zeit, für eine optimale Softwareentwicklung speziell ausgebildete Personen in Prozessmoderationsmethoden für die Analyse- und Bewertungsphase einzusetzen.

3.4 QUADRANT-II: DIE SPEZIFIKATIONSPHASE

Im Anschluss an die Analyse und Bewertung des Arbeitssystems gilt es die gewonnenen Ergebnisse in eine implementierungsnahe Form zu transformieren. Hierzu sind Spezifikationsmethoden mit hohem kommunikativem Wert einzusetzen (z.B. 'RFA'-Netze von Oberquelle 1987 und andere Verfahren: Martin und McClure 1985, Martin 1988, Wood und Silver 1989, August 1991, Raasch 1991).

Die Spezifikation der Werkzeugschnittstelle:

Bei der Spezifikation der Werkzeugschnittstelle wird die intendierte Festlegung zur Mensch-Computer-Funktionsteilung vorgenommen; insbesondere geht es darum, die für die Aufgabenbearbeitung entscheidungsrelevanten Informationen und die dazu notwendigen Arbeitsobjekte mit den entsprechenden Funktionen herauszufinden (Diaper 1989; Dunckel et al. 1993; siehe auch Übersicht 3 im Anhang). Die Aufgaben, welche beim Menschen verbleiben, sollten sich durch die in Teil A, Kapitel 1.2 aufgeführten Merkmale der Aufgabengestaltung auszeichnen.

Die Spezifikation der Ein-/Ausgabeschnittstelle:

Nachdem (einigermassen) Klarheit unter allen Beteiligten darüber besteht, welche Funktionen automatisiert werden, empfiehlt es sich, zunächst mit den Endbenutzern das Bildschirmlayout, z.B. mittels Handskizzenentwurf, zu entwerfen und auszutesten. Bei sehr umfangreichen Mengen unterschiedlicher Masken kann eine Bilddatenbank die mittels Grafikeditor erstellten Masken verwalten. Der Einsatz von Prototypingwerkzeugen für diesen Zweck ist oft deshalb unangemessen, weil die 'tool'-spezifischen

Darstellungsmöglichkeiten oft zu begrenzt sind. Die Auswirkungen der umgesetzten Gestaltungsmassnahmen lassen sich durch Expertenevaluation, Diskussionen mit den Endbenutzern, z.B. im Rahmen von Workshops bzw. mittels Check-Listen (siehe Übersicht 2 im Anhang), austesten, überarbeiten und optimieren.

Die Spezifikation der Dialogschnittstelle:

Zur Spezifikation der Dialogschnittstelle ist es unabdingbar, Prototypen zur Veranschaulichung der dynamischen, interaktiven Aspekte des zu entwikkelnden Werkzeuges einzusetzen. Prototypen sollten nur ganz gezielt und zweckgebunden zur Abklärung spezieller Spezifikationsaspekte eingesetzt werden, weil ansonsten die Gefahr besteht, zuviel Aufwand in den Ausbau und den Erhalt von 'Anschauungsprodukten' zu investieren. Eine sehr effiziente und zudem preiswerte Variante ist die Verwendung von Simulationsstudien, z.B. mittels handgemalter Folien usw., welche dem Benutzer in Abhängigkeit von seinen Aktionen von einem Testleiter, welcher das geplante Systemverhalten ausreichend kennt, vorgelegt werden. Die Auswirkungen der umgesetzten Gestaltungsmassnahmen lassen sich z.B. mittels spezifischer Tests (siehe Teil B, Kapitel 4) oder mit Checklisten (siehe Übersicht 2 im Anhang) austesten, überarbeiten und optimieren.

3.5 QUADRANT-III: DIE REALISIERUNGSPHASE

Nachdem in der Analyse- und Spezifikationsphase der notwendige Optimierungsaufwand geleistet wurde, kann die Realisierungsphase beginnen. Die Realisierungsphase gliedert sich in die folgenden drei Schritte:

(1) Entwurf der Programm-Architektur; (2) Entwurf der einzelnen Programm-Module (-Objektklassen usw.); (3) Codierung und 'Debugging'.

Es ist wichtig, den Entwurf von der Spezifikation zu unterscheiden. Während in der Spezifikation möglichst präzise alle relevanten Eigenschaften des technischen Teilsystems festgelegt werden, muss in der Realisierungsphase dafür Sorge getragen werden, dass das zu entwickelnde technische Teilsystem möglichst alle diese Eigenschaften aufweist. Hier kommen primär rein softwaretechnische Kenntnisse zum Tragen: einfache und klare Systemstruktur, Entkopplung von Bausteinen, Minimalität und Abstraktheit von Schnittstellen, Vermeidung globaler Daten, Bildung abstrakter Systemsichten usw. (Sommerville 1989; Ott 1991; Frühauf, Ludewig und Sandmayr 1991a).

Vor der eigentlichen Codierungsphase ist zu prüfen, inwieweit sich schon vorhandene Software wiederverwenden lässt. Sollte dies nur beschränkt oder gar nicht möglich sein, ist zu testen, welche Programmier- bzw. Entwicklungsumgebung ('CASE workbench') die grösstmögliche Effizienz verspricht. Zu einer Entwicklungsumgebung gehört zunächst im Zentrum ein Sammelbehälter ('information repository'). In diesen Sammelbehälter werden Dokumente und Produkte mittels verschiedener Werkzeuge abgelegt bzw. entnommen: das 'data dictionary', Werkzeuge zum Suchen im Sammelbehälter, Werkzeuge zur Erzeugung strukturierter Diagramme, Werkzeuge zur Entwurfsanalyse und -testung, entwurfsgesteuerte Programmgeneratoren, Maskengeneratoren, Reportgeneratoren, Import- und Exportmöglichkeiten. Durch den Einsatz derartiger Entwicklungsumgebungen lassen sich Produktivitätssteigerungen bis zu 40% erreichen (Chikofsky und Rubenstein 1988).

Die einzelnen Programmteile bzw. die integrierte Version lassen sich mittels Alpha- und Beta-Tests hinsichtlich Korrektheit, Performanz usw. austesten (siehe Frühauf, Ludewig und Sandmayr 1991b).

3.6 QUADRANT-IV: DIE IMPLEMENTATIONS- UND BENUTZUNGSPHASE

Nach Erstellung einer lauffähigen Version kann diese zu Benutzbarkeitsstudien (benutzungsorientierte Benutzungstests, Pilot-Benutzer) im konkreten Arbeits- und Aufgabenkontext eingesetzt werden. Erst zu diesem Zeitpunkt lassen sich alle Probleme mit dem aktuellen organisatorischen und technischen Umfeld abklären. Diese Feldstudien tragen im Unterschied zu Laborstudien dem Aspekt der 'ökologischen Validität' Rechnung. Durch die konkrete Benutzung im realen Aufgabenkontext kann der erreichte Anpassungsgrad an den geplanten organisatorischen Soll-Zustand mittels Interview, Fragebogen, Benutzungstest usw. überprüft und getestet werden. Daten über die Benutzung lassen sich auf Video aufnehmen, durch den Testleiter protokollarisch festhalten oder automatisch in Logfiles abspeichern. Obwohl Videoaufnahmen die informationsreichsten Datenaufzeichnungen sind, hat sich eine Kombination aus Logfile und direkter Protokollierung als ein guter "Kompromiss zwischen Leistungsfähigkeit und Ökonomie" herausgestellt (Müller-Holz et al. 1991).

3.7 DAS ITERATIV-ZYKLISCHE ABLAUFMODELL

Die beim Start und beim Durchlauf einzelner Optimierungszyklen zu beachtenden Aspekte werden im folgenden diskutiert (siehe auch Hesse, Merbeth und Frölich 1992). Als eine wesentliche Rahmenbedingung von Softwareentwicklung hat sich z.B. der Typ der zu entwickelnden Software herausgestellt (Typ A, B, C, D; siehe Teil B Kapitel 2.2).

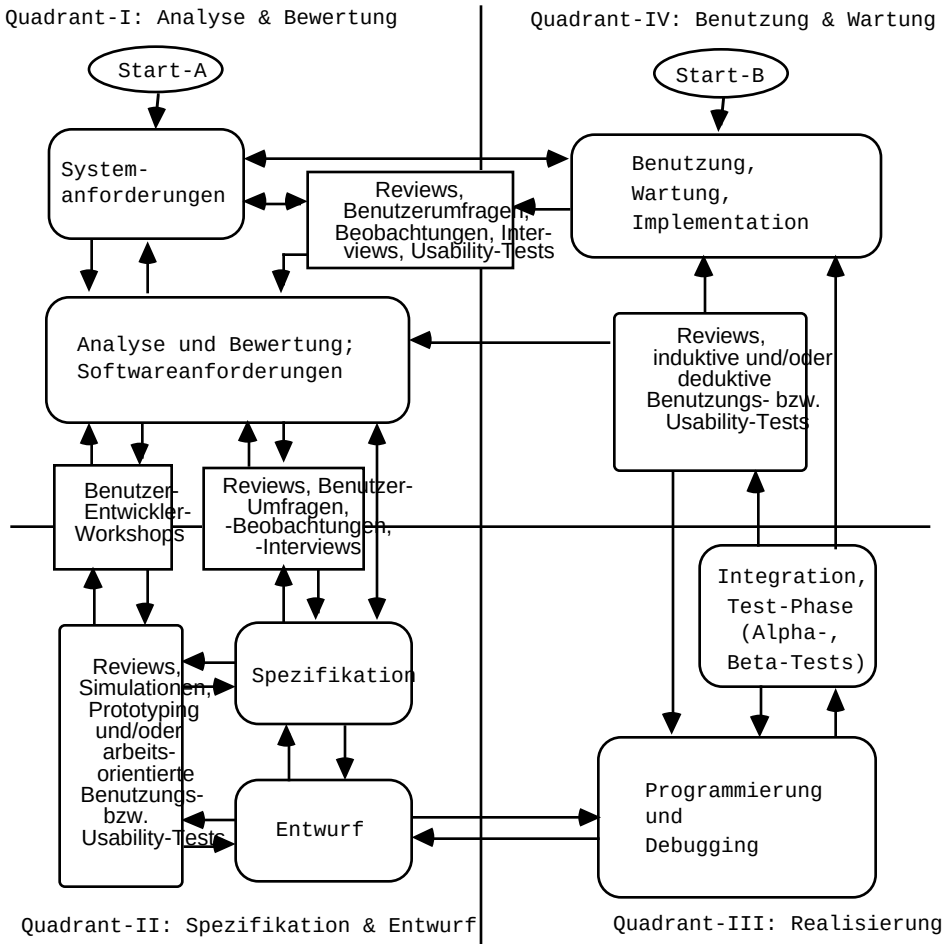


Abbildung 12: Ablaufmodell eines partizipativen Softwareentwicklungskonzeptes mit einer Übersicht über die verschiedenen Optimierungszyklen innerhalb und zwischen den einzelnen Quadranten.

Der Einstieg in den globalen Optimierungszyklus (siehe Abbildung 12) bei einer Neuentwicklung erfolgt über den Start-A, während bei einer Weiterentwicklung der Einstieg bei Start-B erfolgt. Je nach Projekttyp kommen kontextspezifisch unterschiedliche lokale Optimierungszyklen zur Optimierung spezifischer Arbeitsergebnisse zum Einsatz. Die Entscheidung über die jeweils aktuelle Vorgehensweise gehört zur Aufgabe des Projektmanagements und schlägt sich in der gewählten Aufbauorganisation nieder (vergleiche Teil B, Kapitel 2).

Der globale Optimierungszyklus mit seinen eingebetteten lokalen Zyklen lässt sich in vier Bereiche unterteilen (Quadrant-I bis -IV, siehe Abbildung 12). Quadrant-I umfasst die Analyse und Bewertung des Arbeitssystems, in dem die Software eingesetzt werden soll. Hierbei werden überwiegend kommunikative, informale bzw. semiformale Methoden der Arbeits- und Organisationspsychologie verwendet. Im Quadrant-II wird die (Detail-) Spezifikation z.B. unter Einsatz von Prototypen optimiert. In diesem zweiten Quadranten werden verstärkt formale Methoden eingesetzt.

Im Quadrant-III wird die spezifizierte und entworfene Software programmiert und an Testdaten überprüft. Der Quadrant-IV umfasst die Testung und Optimierung des entwickelten Systems in der realen Einsatzumgebung. Die Rückkopplung von der Benutzungsphase (Quadrant-IV) zur Analyse- und Bewertungsphase (Quadrant-I) ermöglicht ein Vorgehen nach dem Versionenkonzept.

Das traditionelle Phasenmodell ist als Spezialfall in dem hier vorgestellten Ablaufmodell enthalten: Der Softwareentwicklungsprozess verläuft weitgehend ohne Rückkopplungen von Start-A über die System- und Softwareanforderungen, die Spezifikationsphase in den Entwurf und die Programmierung; danach wird das Softwaresystem ausgetestet und im Arbeitssystem implementiert. Ohne Rückkopplung zum Quadrant-I wird das Projekt beendet.

Je nach Projekttyp und -auftrag wird der Optimierungsaufwand in den einzelnen Quadranten unterschiedlich ausfallen. Alle bisher durchgeführten Analysen von Softwareentwicklungsprojekten zeigen jedoch, dass mit zunehmendem Optimierungsaufwand in Quadrant-I der Wartungsaufwand in Quadrant-IV abnimmt und insgesamt Kosten gespart werden. Je weniger Aufwand jedoch in Quadrant-I investiert wird, desto mehr Anpassungs- bzw. Wartungsaufwand fällt in Quadrant-IV an. Dieser seit über zehn Jahren bekannte Zusammenhang ist nach wie vor gültig. Tatsächlich ist es eine irrige Vorstellung, man könne die ungelösten Probleme des ersten

Quadranten ohne weiteres im vierten Quadranten aufarbeiten (Boehm 1981, S. 40).

4 Vorgehensweisen und Methoden in der Praxis

4.1 ÜBERSICHT

Wie eingangs im letzten Kapitel vor dem Hintergrund systemtheoretischer Betrachtungen aufgezeigt wurde, setzt sich jeder einzelne Optimierungszyklus aus einer Test- und einer Aktivitätskomponente mit entsprechender Verkopplung zusammen. Beide Komponenten können recht unterschiedlicher Art sein. In der Tabelle 5 geben wir eine Übersicht über Einsatzschwerpunkte, Arten der Aktivität und Tests, das jeweilige Arbeitsergebnis sowie den geschätzten Bereich der Zyklusdauer. Je kürzer die Zyklusdauer ausfällt, desto schneller und damit öfter kann der Optimierungszyklus durchlaufen werden. Je öfter ein Optimierungszyklus durchlaufen wird, desto eher kann ein optimales Ergebnis erreicht werden.

Für die Festlegung des Zeitrahmens der einzelnen Aktivitäten kann man nun anhand der Zyklusdauer der einzelnen Optimierungszyklen ausrechnen, wieviel Optimierung zu welchem Preis machbar und ggf. erstrebenswert ist. Sind bei der Projektetablierung die benötigten Optimierungszyklen nach inhaltlichen Gesichtspunkten festgelegt, so kann man ausrechnen, wieviel Zeit in etwa für die einzelnen Optimierungsaktivitäten benötigt wird: nämlich mindestens die Dauer des ersten Zyklus. Wieviele Zyklusdurchläufe notwendig werden, hängt von den Ergebnissen der jeweiligen Testphase ab; ab zwei Zyklusdurchläufen kann man schon abschätzen, wie schnell die 'gemessene' Produktqualität sich nach diesen zwei Optimierungszyklen dem wahrscheinlich noch nicht erreichten Soll-Wert annähert.

Ein wesentliches Problem der adäquaten Verzahnung der verschiedenen Optimierungszyklen besteht in ihrer *Synchronisation*. Sind an verschiedenen Stellen im Quadrantenmodell (siehe Abbildung 12) gleichzeitig mehrere Optimierungszyklen aktiv, so müssen diese adäquat synchronisiert werden. Dieser Aspekt ist deshalb besonders wichtig, weil nur so das Ausmass an Inkonsistenzen innerhalb des gesamten Entwicklungsprozesses minimiert werden kann. Wenn z.B. parallel zur Realisierungsphase weitere Rücksprachen und Anforderungsanalysen mit dem Anwender stattfinden, passiert es leicht, dass die Entwickler gemäss den stets veralteten Spezifikationen oftmals für den 'Papierkorb' programmieren. Die Ursache hierfür ist

bei fehlender oder mangelnder Synchronisation dieser beiden Optimierungszyklen in ihrer unterschiedlichen Zyklusdauer zu sehen.

Methode	Aktivität	'Test'	Ergebnis	Zyklus-Dauer
Diskussion (siehe Teil B, Kapitel 4.5)	verbale Kommunikation Metaplan, Flipcharts usw.	rein verbale Interpretation visuelle + verbale Interpretation	globale Design-Entscheidungen spezifische Design-Entscheidungen	Sekunden – Minuten Minuten – Stunden
Simulation (siehe Teil B, Kapitel 4.8)	Handskizzen, Szenarien, 'wizard of oz' usw. Erstellung von Ablaufplänen usw. mittels (semi-)formaler Methoden	visuelle + verbale Interpretation visuelle + verbale Interpretation bei entsprechender Qualifikation	Spezifikation der Ein-/Ausgabeschnittstelle (semi-)formale Beschreibungs-Dokumente	Minuten – Tage Stunden – Wochen
Prototyping (siehe Teil B, Kapitel 4.10)	horizontales Prototyping partiell vertikales Prototyping vollständig vertikales Prototyping	'lautes Denken', 'walk-through' heuristische Evaluation aufgabenorientierte Benutzungs-, Usability-Tests	Spezifikation der Dialogkomponente Spezifikation von Teilen der Anwendungskomponente Spezifikation der Anwendungskomponente	Tage – Wochen Wochen – Monate Wochen – Monate
Versioning	erster Durchlauf des gesamten Entwicklungszyklus nächster Durchlauf des gesamten Entwicklungszyklus	induktive Benutzungs-, Usability-Tests deduktive Benutzungs, Usability-Tests	erste weitgehend vollständige Version mehrere weitgehend vollständige Versionen	Monate – Jahre Monate – Jahre

Unter dem *divisionalen Synchronisationsprinzip* verstehen wir die Aufteilung des insgesamt zu erstellenden Funktionsumfanges des Anwendungs-

systems in relativ unabhängige, produktorientierte Bereiche. Durch diese Aufteilung wird der Verkopplungsgrad zwischen den verschiedenen Optimierungszyklen minimiert. Je unabhängiger im Gesamtprojekt die einzelnen produktorientierten Bereiche in der Organisationsgestaltung konzipiert werden, desto wichtiger ist es, zu Beginn des Projektes eine möglichst optimale Aufteilung zwischen den Funktionsbereichen anzustreben. Hieran kann man erkennen, dass Softwaregestaltung sehr eng an Arbeits- und Organisationsgestaltung gekoppelt ist. Da jedoch die Trennung der einzelnen Funktionsbereiche des Anwendungssystems niemals in vollständig unabhängige Bereiche erfolgen kann, müssen weitere Synchronisationsprinzipien zum Einsatz kommen.

Durch das *informationelle Synchronisationsprinzip* wird eine informationelle Kopplung zwischen den einzelnen Optimierungszyklen aufgebaut, sodass alle Personen in den unterschiedlichen Zyklen über den aktuellen Stand in den parallel aktiven Zyklen informiert sind. Dies kann durch einfache Hilfsmittel wie Aktenordner an einem definierten Standort, durch regelmässige Besprechungstermine, aber auch mittels technischer Unterstützung (Mailbox, Versionendatenbanken, 'information repository' usw.) realisiert werden.

Das *personelle Synchronisationsprinzip* besteht darin, dass der Personenkreis, welcher an den unterschiedlichen Optimierungszyklen teilnimmt, *derselbe* ist. Ein Projektmitglied, welches z.B. an einer weiteren Sitzung zur Anforderungsanalyse teilnimmt, kann nicht *gleichzeitig* programmieren; hierdurch wird gewährleistet, dass alle oder zumindest ein wesentlicher Teil der Projektmitglieder den aktuellen Wissensstand haben. Dieses Prinzip bricht sich jedoch oftmals an der arbeitsteilig orientierten Organisationsform, welche bei vielen Softwarehäusern noch vorhanden ist. Hier müsste eine Reorganisation dieser Softwareentwicklungsabteilungen vorgenommen werden.

In diesem Kapitel werden verschiedene Methoden zur Benutzerbeteiligung aufgeführt, welche sich an den verschiedenen Stellen des iterativ-zyklischen Ablaufmodells einsetzen lassen. Die Befragungsmethoden sind z.B. so generell einsetzbar, dass sie in verschiedenen Quadranten zum Einsatz kommen können. Die Benutzungs- bzw. Usability-Tests werden ausführlicher als die anderen Methoden dargestellt, weil sie bisher im deutschsprachigen Bereich noch nicht weiter in dieser Form dargestellt worden sind (siehe dazu speziell Nielsen 1993 sowie Dumas und Redish 1993). Die folgende Tabelle 6 kann dabei behilflich sein, die Zuordnung einzelner – im

folgenden genauer erläuterter – Methoden zu den verschiedenen Projektaktivitäten (Analyse, Bewertung, Darstellung, Produktion, Test) vornehmen zu können (weitere Methoden sind in Heilmann 1981, Wicke 1988, Greenbaum und Kyng 1991 sowie Haberfellner et al. 1992 beschrieben). Wir haben an verschiedenen Stellen in diesem Buch dargelegt, dass sich produktive Methoden besonders gut auch und gerade in der Analysephase eignen, um die gemeinsamen Vorstellungen in bezug auf die wichtigen und notwendigen Anforderungen herausarbeiten zu können.

	A	**B**	**D**	**P**	**T**
Arbeitsanalyse mit Raster bzw. standardisierten Erhebungsverfahren (siehe Übersicht 3 im Anhang)	X	X			
Benutzerorientierte Benutzungstests		X			X
Beobachtung (strukturiert), Videoaufnahme	X				X
Comic strips, Paper pencil		X	X	X	
Diagramme (Flowchart, HIPO usw.)	X		X	X	
Dokumentenanalyse (z.B. Stellenbeschreibung)	X	X			
Fragebogen (z.B. Registrierkarte mit Fragebogen)	X	X		X	
Generatoren, Tools (z.B. Maskengenerator)				X	
Interview, Gruppendiskussion	X	X		X	X
Kärtchenmethode (z.B. Metaplan)	X			X	
Kreativitätstechniken (z.B. Brainstorming)				X	
Logfile (Datenschutz!)	X	X			X
Prototyping	X	X	X	X	X
Review, Walk-through		X			X
Szenarien			X	X	X
Workshop	X	X		X	X

4.2 KONZEPTION EINES BETEILIGUNGSMODELLS

Das hier beschriebene Modell der Benutzerbeteiligung setzt nach der Konzeption des Arbeitssystems im ersten Schritt zur Softwareentwicklung ein. Es steht also bereits fest,

(1) wie die organisatorische Struktur des Arbeitssystems aussieht,

(2) welche Aufgaben in diesem Arbeitssystem bearbeitet werden,

(3) welche Aufgaben mit Hilfe des Softwaresystems unterstützt werden sollen.

Bei der Softwareentwicklung im engeren Sinne wird es nun primär um die Frage gehen, wie die Aufgaben am besten mit Software zu unterstützen sind. Unser Modell der Softwareentwicklung integriert für die Bearbeitung dieser Frage die zukünftigen Benutzer in den Softwareentwicklungsprozess. Der Erfolg dieses Vorhabens hängt dabei entscheidend von der Transparenz ab, mit welcher die Benutzerbeteiligung durchgeführt wird. Es kann immer wieder beobachtet werden, dass die Benutzer in Softwareentwicklungsprojekten nicht aktiv mitarbeiten wollen, weil die Beteiligung erst beim Auftreten von Problemen im Entwicklungsprozess ad hoc organisiert wird. Oftmals fehlt den Benutzern in diesen Situationen der Überblick über die Gesamtzusammenhänge im Projekt, stellt sich bei ihnen das – nicht sehr motivierende – Gefühl ein, plötzlich bei der Lösung von Problemsituationen wichtig zu sein, ohne zu deren Entstehung beigetragen zu haben.

Um das Auftreten solcher Schwierigkeiten möglichst zu verhindern, sollte vor Beginn der eigentlichen Projektarbeiten ein Beteiligungsmodell konzipiert werden, welches als Richtlinie für die Benutzerbeteiligung gilt. So wird die Benutzerbeteiligung für alle Betroffenen durchschaubar und voraussehbar, man kann sich darauf einstellen und Einfluss geltend machen. In diesem Kapitel wird deshalb auf der Grundlage der Ausführungen im Teil A dargestellt, wie ein Beteiligungsmodell erstellt werden kann und welche Kernelemente darin berücksichtigt sein müssen.

Ein Beteiligungsmodell ist das Produkt enger planerischer Kooperation zwischen dem technischen Projektleiter, dem Leiter der direkt betroffenen Fachabteilung, Repräsentanten der indirekt betroffenen inner- und ausserbetrieblichen Instanzen und einem – vor allem bei grösseren Projekten empfehlenswert – eigens für die Beteiligungsorganisation Verantwortlichen. In dieser Phase ist auch die Mitarbeit der Arbeitnehmervertretung von entscheidender Bedeutung. Bereits bei der Erstellung des Beteiligungsmodells

ist also die Zusammenarbeit zwischen Vertretern unterschiedlicher Kompetenzbereiche unabdingbar.

Titelblatt	**Beteiligungsmodell** • Projektname • Verfasser • Verteiler • Datum und Version
1. Teil	*Grundlagen und Voraussetzungen* • Merkmale des Arbeitssystems • Zielsetzung und Ablaufstruktur des Projektes • Projektrahmenbedingungen • Technische Rahmenbedingungen • Personelle Besetzung • Zeitrahmen • Betroffenenstruktur • Endbenutzer • Indirekt Betroffene
2. Teil	*Konkretisierung der Beteiligung in den Projektphasen* • Analyse und Bewertung • Spezifikation • Realisierung • Implementation und Benutzung
3. Teil	*Allgemeine Bestimmungen* • Freistellung • Qualifizierungsmassnahmen für Beteiligung • Informationspolitik für Nichtbeteiligte

Das Beteiligungsmodell wird in einem Dokument festgehalten, welches vor Beginn der eigentlichen Entwicklungsarbeiten allen potentiell Betroffenen zugänglich gemacht werden sollte. Es muss deshalb in allgemein verständlicher, nichttechnischer Sprache abgefasst werden und darf nicht zu umfangreich sein. Es sollte drei Teile umfassen:

1. Teil: Grundlagen und Voraussetzungen

2. Teil: Konkretisierung der Beteiligung in den Projektphasen

3. Teil: Allgemeine Bestimmungen

Tabelle 7 gibt einen Überblick über die inhaltliche Grobstruktur des Beteiligungsmodells. Bevor nun die einzelnen Teile des Beteiligungsmodells näher beschrieben werden, müssen zwei Anmerkungen vorangestellt werden:

(1) Es ist schwierig, wenn nicht sogar unmöglich, Softwareentwicklungsprozesse im voraus bis in die kleinste Einzelheit zu planen. Änderungen in den Anforderungen, technische Pannen, Personalfluktuation und finanzielle Probleme stellen nur einige der Gründe dar. Deshalb wäre es falsch, das Beteiligungsmodell von Beginn weg allzu detailliert auszulegen. Die anschliessend beschriebene Form ist deshalb als ein Rahmen zu betrachten, der im Verlauf des konkreten Entwicklungsprojektes kontinuierlich ausgearbeitet und verfeinert wird.

(2) Trotz der oben angesprochenen Dynamik von Softwareentwicklungsprozessen erhält das Beteiligungsmodell seine Bedeutung nur durch eine gewisse Verbindlichkeit, die ihm von allen beteiligten Interessenvertretungen anerkannt werden muss. Es muss in diesem Sinne durchaus einen Bestandteil der offiziellen Projektabmachungen zwischen den Kooperationspartnern bilden.

4.2.1 Grundlagen und Voraussetzungen

Im ersten Teil des Beteiligungsmodells werden die wichtigsten Grundlagen und Voraussetzungen für das Softwareentwicklungsprojekt so beschrieben, dass alle Betroffenen einen Überblick über die Gründe, Zielsetzungen und Rahmenbedingungen des Projektes erhalten. Im einzelnen sind dies

Merkmale des Arbeitssystems

Als Einstieg in das Beteiligungsmodell werden die Ergebnisse des ersten vorbereitenden Schrittes – Analyse, Bewertung und Gestaltung des Arbeitssystems – dargestellt. Besondere Aufmerksamkeit ist dabei der Beschreibung der zukünftigen Form des Arbeitssystems zu widmen. Damit wird erreicht, dass zwischen allen Beteiligten im Softwareentwicklungsprozess Einigkeit über den Zielzustand des Arbeitssystems herrscht und der Einsatzbereich der zukünftigen Softwareanwendung klar ist.

Zielsetzung und Ablaufstruktur des Projektes

Dieser Teil enthält erste Angaben über allgemeine Merkmale der zukünftigen technischen Unterstützung und eine Übersicht über das vorgesehene Vorgehen bei der Entwicklung. Als allgemeine Merkmale gelten einerseits die Art der Rechnerintegration (z.B. Netzwerk, unabhängige Arbeitsplatzrechner), die Art der Anwendung (z.B. Datenbank, Textverarbeitung, Funktionsintegration) sowie softwareergonomische Kriterien, die von besonderer Relevanz sind (z.B. Transparenz, Flexibilität). In der Übersicht über die Vorgehensschritte bei der Entwicklung werden die Beteiligten über die logische Ablaufstruktur des Projektes informiert.

Projektrahmenbedingungen und technische Rahmenbedingungen

Softwareentwicklungsprojekte können selten auf der 'grünen Wiese' gestartet werden. Deshalb sollte von Beginn weg im Beteiligungsmodell dargelegt werden, ob und gegebenenfalls welche Vorgaben bezüglich Hardware und Betriebssystemen, Einbindung bestehender Softwarelösungen und sonstiger technischer Installationen zu berücksichtigen sind.

Personelle Besetzung

Die Beschreibung der personellen Besetzung des Entwicklungsprojektes gibt den Beteiligten Auskunft über die Aufgabenzuordnung im Softwareentwicklungsteam. Mit diesen Angaben wissen die Beteiligten, mit wem sie es im konkreten Fall zu tun haben, und können mit Fragen, Unklarheiten und Ideen direkt an die entsprechenden Personen gelangen.

Zeitrahmen

Die Kenntnis des zeitlichen Rahmens für die Softwareentwicklung ermöglicht es den Betroffenen, einerseits ihren Einsatz als Beteiligte vorauszuplanen und andererseits realistische Erwartungen für den Zeitpunkt der Inbetriebnahme der Anwendung zu haben. Allfällige Verzögerungen im Projektablauf können anhand dieses Referenzzeitrahmens transparent und verständlich dargestellt werden.

Betroffenenstruktur und Endbenutzer

Damit bei der Durchführung der Benutzerbeteiligung in den verschiedenen Phasen des Projektablaufes die Repräsentativität eingehalten werden kann, lohnt sich die Erstellung einer Übersicht über die Betroffenen. Vor allem

für die Gruppe der zukünftigen Endbenutzer sollte die Übersicht über die folgenden Benutzermerkmale Auskunft geben:

(1) Vorgesehene Aufgaben im neuen Arbeitssystem,

(2) hierarchische Stellung im Betrieb,

(3) Beruf, Ausbildung,

(4) Computererfahrung,

(5) Sprache,

(6) Alter,

(7) Dauer der Betriebszugehörigkeit.

Aufgrund dieser Benutzermerkmale können für die Projektarbeit Benutzergruppen mit jenen Merkmalsausprägungen zusammengestellt werden, die der anstehenden Problemstellung am besten entsprechen. Die Benutzermerkmale können vor allem in grossen und geografisch dezentralen Unternehmen mittels eines einfachen Fragebogens, der von den zukünftigen Benutzern selbst ausgefüllt wird, erhoben werden.

Indirekt Betroffene

Indirekt betroffen sind vor- und nachgelagerte Betriebsbereiche, Kunden und Personen, die selbst nicht mit der zukünftigen Anwendung arbeiten werden, aber von deren Einsatz durch Veränderungen in ihrem eigenen Bereich betroffen sind. Die indirekt Betroffenen sind deshalb vor allem bei Fragen der Schnittstellengestaltung und Systemabgrenzung gegen aussen in den Entwicklungsprozess einzubeziehen. Für die Konzeption der Benutzerbeteiligung genügt in der Regel die Aufzählung der verschiedenen Gruppen, ohne dass detaillierte Merkmalsausprägungen wie bei den Endbenutzern spezifiziert werden müssen.

4.2.2 Konkretisierung der Beteiligung in den Projektphasen

Nach dem Gesamtüberblick im ersten Teil wird im zweiten Teil des Beteiligungsmodells die geplante Benutzerbeteiligung für jede der angegebenen Projektphasen dargestellt. Der Detaillierungsgrad hängt dabei stark von der Projektgrösse ab. Vor allem bei Projekten, welche eine Vielzahl unterschiedlicher Aufgaben und Benutzer betreffen, ist es wichtig, das Beteili-

gungsmodell zu diesem Zeitpunkt noch nicht allzu detailliert zu konzipieren. Folgende Dimensionen sollten aber zumindest grob umrissen werden.

Erarbeitung und Evaluation

Beteiligungsaktivitäten unterteilen sich in solche, bei welchen Lösungen konstruktiv erarbeitet werden, und solche, bei welchen erarbeitete Lösungen geprüft, bewertet und gegebenenfalls überarbeitet werden. Diese Trennung ist vor allem vor dem Hintergrund iterativ-zyklischer Prozessabläufe bedeutsam. Je nach Aufgabe und Struktur der Benutzergruppe kann es sinnvoll sein, für die Erarbeitung und Evaluation unterschiedliche Personen zu beteiligen. Durch die personell getrennte Durchführung können Lösungen auf einer breiteren Basis abgestützt werden.

Direkt – indirekt

Die Unterscheidung zwischen direkter und indirekter Beteiligung bezieht sich auf die Form der Kommunikation und Kooperation zwischen Softwareentwicklern und Beteiligten. Direkte Beteiligung ist gegeben, wenn Kommunikation und Kooperation in unmittelbarer Interaktion vorliegt. Indirekte Beteiligungsformen beschränken sich auf schriftliche Erhebungen, Vertretung der Endbenutzer durch Personen, die im zukünftigen Arbeitssystem nicht selbst die technisch unterstützten Aufgaben wahrnehmen (z.B. Vorgesetzte, Organisatoren usw.).

Inhalt

Der Inhalt der Beteiligung bezieht sich auf den Gegenstand, der bearbeitet wird. In der Phase der Analyse und Bewertung bilden die Datenstruktur, die Aufgabenstrukturen und die Systemfunktionalität der einzelnen zu unterstützenden Aufgaben die zentralen Inhalte. In den übrigen Phasen sind die Ein-/Ausgabeschnittstelle sowie die Dialogschnittstelle von zentralem Interesse. Im Interesse der Klarheit muss aus dem Beteiligungsmodell klar ersichtlich sein, auf welche Aufgaben oder Teilaufgaben sich ein Inhalt bezieht.

Die Informationen zu den Aufgaben 1 und 2 in der Abbildung 13 könnten im Rahmen eines Softwareentwicklungsprojektes für eine Schulverwaltung beispielsweise folgendes bedeuten

Aufgabe 1: Fortlaufendes Erfassen der Prüfungsnoten der Schüler während eines Semesters.

Aufgabe 2: Berechnen des Notendurchschnittes am Semesterende.

Daten: Inhalt ist die vollständige Erfassung und Strukturierung aller mit diesen Aufgaben verbundenen Daten und ihrer Merkmale (Lehrer, Fach, Schülername usw.).

Methode

Für jeden Inhalt wird in Abhängigkeit der Dimensionen 'Erarbeitung - Evaluation' und 'direkt - indirekt' die Methode genannt, die bei der Benutzerbeteiligung angeführt wird. Hier eignen sich zum besseren Verständnis für die Benutzer allgemein formulierte Begriffe wie Workshop, Fragebogen, Interview.

Benutzergruppe

Für jeden Inhalt werden aufgrund der Betroffenenstruktur, wie sie im ersten Teil des Beteiligungsmodells aufgeführt ist, die Benutzergruppen repräsentativ zusammengestellt. Die Merkmalsausprägungen für jede Benutzergruppe (z.B. Sprache, Ausbildung, zukünftige Aufgaben, EDV-Erfahrung) sind so darzustellen, dass sich an der Beteiligung Interessierte selbst einordnen können. Aufgrund dieser Einordnung und der persönlichen Interessen bezüglich Inhalte, Erarbeitung, Evaluation, Methode und zeitlicher Lage besteht somit für jeden Betroffenen die Möglichkeit, sich selbst aktiv für eine Beteiligung am Softwareentwicklungsprozess zu melden. Konkret können die Benutzergruppen wie folgt zusammengesetzt sein:

Gruppe A: Sachbearbeiter der verschiedenen Abteilungen,

Gruppe C: Lehrer,

Gruppe D: Schulleitung, Ebene Abteilungsleitung,

Gruppe F: Schülervertretung.

Abfolge

Die Abfolge gibt das zeitliche Aufeinanderfolgen der einzelnen Aktivitäten an. Damit werden iterativ-zyklische Vorgehensweisen transparent gemacht. Gleichzeitig geplante Aktivitäten erhalten die gleiche Zahl in der Reihenfolge. Die Planbarkeit für die Beteiligten erhöht sich entscheidend, wenn nebst der Reihenfolge auch bereits der Zeitraum angegeben werden kann, für welchen eine Beteiligungsaktivität geplant ist (z.B. Projektmonate 5, 6,

7). Für die Analyse der Datenstruktur der Aufgaben 1 und 2 sieht das Beteiligungsmodell also folgendes Vorgehen vor:

(1) In einem ersten Schritt (Abfolge Nr. 1) werden bei den Sachbearbeitern, Lehrern und den Abteilungsleitern Informationen über die benötigten Daten für Aufgabe 1 und 2 mit einem Fragebogen erhoben. Während der Fragebogenaktion findet ein Workshop statt, in welchem interessierte Sachbearbeiter und Lehrer zusammen mit dem Softwareentwickler erste Überlegungen zur Datenstruktur anstellen.

Ergebnis dieser Aktivitäten ist eine schriftliche Datenliste, in welcher der Softwareentwickler die Daten und Merkmalsausprägungen festhält.

(2) Diese Datenliste wird in einem zweiten Schritt (Abfolge Nr. 2) von den Sachbearbeitern, Lehrern und Abteilungsleitern durchgesehen und korrigiert.

(3) In einem dritten Schritt (Abfolge Nr. 3) erarbeiten nun Sachbearbeiter, Lehrer, Abteilungsleiter und auch ein Vertreter der Schülerschaft zusammen mit dem Softwareentwickler das vollständige und korrekte Datenmodell für die Bearbeitung dieser Aufgaben.

Bei diesem Vorgehen ist es Aufgabe des Softwareentwicklers, die Datenmodelle der einzelnen Aufgaben im Hinblick auf das Gesamtdatenmodell der Softwareanwendung hin zu koordinieren und allfällige Inkompatibilitäten im Verlauf des weiteren Vorgehens zur Diskussion zu stellen.

Es ist selbstverständlich, dass die Beteiligung der Endbenutzer gerade in grossen Projekten nicht in diesem Detaillierungsgrad vorausplanbar ist. Das Modell unterstützt jedoch auch die laufende Planung und Kontrolle während der Projektarbeit, weist auf Unterlassungen hin und hilft, die einzelnen Aktivitäten zu koordinieren. So können zum Beispiel in einem einzelnen Workshop mit einer bestimmten Benutzergruppe verschiedenste Inhalte bearbeitet werden. Mit der Anwendung eines solchen Modelles soll erreicht werden, dass

(1) der Beteiligungsprozess – als ganzer – für alle transparent wird;

(2) dadurch die Möglichkeit geboten wird, persönliche Beteiligungswünsche zu formulieren.

Abbildung 13 zeigt, wie die verschiedenen Dimensionen schematisch dargestellt werden können. Die Übernahme der Dimensionen und Informationen in eine Datenbank ermöglicht es zudem, während des Beteiligungspro-

zesses Informationen gezielt zusammenzustellen und abzufragen. So können z.B. sämtliche Beteiligungsaktivitäten zu einem bestimmten Inhalt oder für eine bestimmte Benutzergruppe übersichtlich kontrolliert und kommuniziert werden.

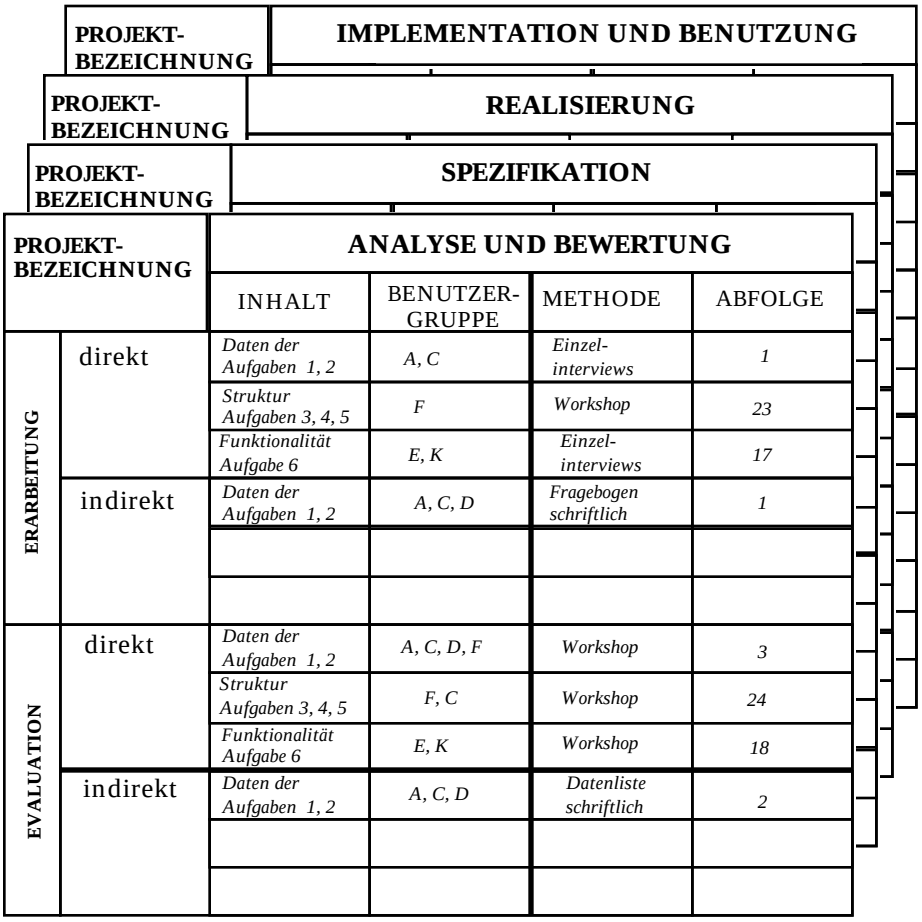


Abbildung 13: Schematische Auslegung der Benutzerbeteiligung.

4.2.3 Allgemeine Bestimmungen

Der dritte und letzte Teil des Beteiligungsmodells enthält allgemeine Bestimmungen, die eine möglichst konfliktfreie und reibungslose Durchführung der Benutzerbeteiligung unterstützen. Zahlreiche Erfahrungen in den verschiedensten Projekten zeigen immer wieder, dass Beteiligungsprozesse wegen unklarer Abmachungen und Kompetenzabgrenzungen schwierig werden und dadurch die Motivation aller involvierten Stellen stark abnimmt. Wir empfehlen deshalb, primär die folgenden drei Bereiche über klare Abmachungen zu regeln.

Freistellung

Insbesondere bei Formen der direkten Benutzerbeteiligung, bei welchen Mitarbeiter der Fachabteilungen regelmässig für Analysen, Workshops oder über bestimmte Zeiträume hinweg der Softwareentwicklung zur Verfügung stehen müssen, ist geregelte Entlastung vom Alltagsgeschäft im Betrieb unabdingbare Notwendigkeit. Leider versuchen Linienvorgesetzte immer wieder, gerade jene Mitarbeiter freizustellen, auf die wegen fehlender Fachqualifikationen oder kurzer Betriebszugehörigkeit im Alltagsgeschäft der Fachabteilung eher verzichtet werden kann. Diese Personen bringen denn auch oft nicht das notwendige Wissen in den Softwareentwicklungsprozess ein, so dass gute und innovative Lösungen schon bald gefährdet sein können.

Qualifizierungsmassnahmen für die Beteiligung

Die aktive und direkte Mitarbeit in Softwareentwicklungsprozessen setzt sowohl bei den Softwareentwicklern als auch bei den Beteiligten selbst gewisse soziale und fachliche Qualifikationen voraus. Vor allem beim Einstieg in die Softwareentwicklung mit direkter Benutzerbeteiligung ist deshalb darauf zu achten, dass diese Qualifikationen vor Projektbeginn vermittelt werden.

Informationspolitik für Nichtbeteiligte

Oftmals ist es gerade bei Entwicklungsprojekten mit einer grossen Anzahl von Betroffenen unmöglich, dass alle Betroffenen direkt oder indirekt beteiligt werden. Die Nichtbeteiligten sind deshalb regelmässig über Projektverlauf, aktuellen Stand und Entscheidungen zu informieren.

4.3 ANALYSE, BEWERTUNG UND GESTALTUNG BESTEHENDER ARBEITSSYSTEME

Bei der Entwicklung und Einführung eines Softwaresystems sind Fragen der Arbeitsorganisation, der Mensch-Computer-Funktionsteilung, der Softwaregestaltung und der Hardwaregestaltung zu beachten. In diesem Kapitel zur Arbeitsorganisation gehen wir davon aus, dass die Arbeit dem Menschen sinnvolle Tätigkeiten und Möglichkeiten zur persönlichen Entfaltung bietet und aus sich selbst heraus motivierend wirkt. Im wesentlichen können folgende Vorteile erwartet werden, wenn man sich vorgängig zur Softwareentwicklung mit den arbeitsorganisatorischen Aspekten auseinandersetzt:

(1) Der optionale Charakter der neuen Technologien kann adäquat genutzt werden. Eine bestehende Arbeitsorganisation wird nicht unreflektiert übernommen und durch die neuen Technologien zementiert.

(2) Mit der Software wird nicht nur ein isoliertes Problem gelöst, welches eventuell nur Symptom, nicht aber Ursache einer unbefriedigenden Situation darstellt.

(3) Neue Technologien werden nicht allein aufgrund ihrer Verfügbarkeit eingeführt, sondern gezielt auf die Verbesserung der Situation für die Betroffenen und die Effizienz des gesamten Arbeitssystems hin entwickelt.

(4) Es entsteht sowohl für die zukünftigen Benutzer als auch für die Softwareentwickler von Beginn an ein gemeinsames Verständnis für das gesamte Arbeitssystem. Dadurch können Anforderungen besser definiert werden. Unsicherheiten, Missverständnissen und Widerstand kann vorgebeugt werden.

Besonders letzteres muss für die Softwareentwickler von grossem Interesse sein. Es kann nämlich immer wieder beobachtet werden, dass Computerbenutzer letztlich gar nicht wegen der Unzulänglichkeit der Software unzufrieden sind, sondern wegen unbefriedigender Arbeitsstrukturen, welche durch die Software unterstützt werden. Damit wird deutlich, dass die Gestaltung der Arbeitssituation nicht primär eine Aufgabe der Softwareentwicklung sein kann, sondern eine Aufgabe der Arbeits- und Organisationsfachleute zusammen mit den Beschäftigten des Arbeitssystems. Den Informatikfachleuten fällt hierbei jedoch die wichtige Aufgabe zu, den Prozess der Arbeitsstrukturierung mit ihrem technischen Wissen zu unterstützen, in-

dem sie technische Möglichkeiten und auch Grenzen aufzeigen. Nur so kann von Beginn an die Forderung nach soziotechnisch optimierten Arbeitssystemen eingelöst werden. Gleichzeitig erhalten die Informatikspezialisten durch ihre Teilnahme an diesen Gestaltungsprozessen viel Hintergrundwissen zu den Überlegungen der zukünftigen Anwender, können Anforderungen an das Softwaresystem besser nachvollziehen und achten bei der Softwaregestaltung bewusster darauf, arbeitsorganisatorische Optionen nicht unnötig technisch einzuschränken.

Obwohl sich dieses Buch nicht primär mit Fragen der Arbeitsstrukturierung und Organisationsentwicklung befasst, möchten wir an dieser Stelle einige zentrale Aspekte aus diesem Bereich darstellen. Ohne im Detail auf die Methoden und Konzepte der Organisationsentwicklung einzugehen – dafür verweisen wir auf die umfangreiche Fachliteratur –, werden hier prozessbezogene und methodische Merkmale von Veränderungsmassnahmen erläutert.

Organisationsentwicklungen sind grundsätzlich als Veränderungsprozesse zu begreifen, in welchen ein unbefriedigender Ausgangszustand (Ist-Zustand) eines Arbeitssystems zusammen mit den Betroffenen in einen optimalen Zielzustand (Soll-Zustand) überführt werden soll. Da der Zielzustand zu Beginn eines Veränderungsprozesses meist nur in Form allgemeiner Vorgaben durch die Geschäftsleitung bekannt ist – z.B. mehr Effizienz, grössere Flexibilität, bessere Kundenorientierung, höhere Motivation –, besteht bei den Betroffenen meist eine erhebliche Unsicherheit bezüglich der konkreten Ziele und Realisierungsschritte. Um diese Unsicherheiten abzubauen und grösstmögliche Transparenz und Identifikation mit dem anzustrebenden Zielzustand zu erreichen, ist auch hier wie in der Softwareentwicklung selbst die aktive Beteiligung aller Betroffenen eine wichtige Voraussetzung für das Gelingen der Veränderungsprozesse. Für die Erarbeitung von arbeitsorganisatorischen Vorgaben für die Softwareentwicklung sollte man sich deshalb an ein Vorgehen halten, welches allen Betroffenen erlaubt,

(1) die Ausgangssituation gemeinsam zu beurteilen,

(2) an der Zielfindung für die Zukunft mitzuarbeiten und

(3) daraus Anforderungen und Gestaltungsziele für das zukünftige Arbeitssystem abzuleiten.

Ein solches Vorgehen hat notwendigerweise drei Schritte zum Inhalt:

Schritt 1: Analyse der vorliegenden Situation.

Schritt 2: Definition der Anforderungen an das zukünftige Arbeitssystem.

Schritt 3: Konzeption des neuen Arbeitssystems.

4.3.1 Schritt 1: Analyse der vorliegenden Situation

Ziel:

Die Analyse der vorliegenden Situation bedeutet eine möglichst umfassende Betrachtung der von einer technologischen Innovation betroffenen Organisationseinheit. Die Herausarbeitung der Stark- und Schwachstellen im bestehenden System steht dabei im Vordergrund. Die Analyseergebnisse sollen Aussagen zur Umweltsituation und zur Aufbau- und Ablauforganisation der Organisationseinheit sowie zum technischen und sozialen Teilsystem des Arbeitssystems enthalten (siehe Übersicht 3 im Anhang).

Methodische Zugänge:

Den wohl umfassendsten Zugang zur Analyse der aktuellen Situation bietet die soziotechnische Systemanalyse. Sie berücksichtigt alle in der Zielstellung genannten Dimensionen und erfasst sie in mehreren aufbauenden Analyseschritten. Ausgehend von einer Grobanalyse des Arbeitssystems, in welcher die formale Organisationsstruktur, die wichtigsten Input- und Outputgrössen, deren Transformationsprozesse sowie ökonomische und soziale Ziele beschrieben werden, werden die folgenden zentralen Fragen zu beantworten gesucht.

Fragen zur Umweltsituation:

- Welche Produkte werden heute und in Zukunft vom Markt verlangt?
- Welche Anforderungen stellt die Umwelt an den Produktionsprozess bezüglich Termine, Qualität, Flexibilität usw.?
- Wie sieht die Konkurrenzsituation aus?

Fragen zu Auftrag und Abhängigkeiten:

- Wie sieht der Gesamtauftrag des Arbeitssystems aus?
- In welchen Beziehungen und Abhängigkeiten zu vor- und nachgelagerten Bereichen befindet sich das Arbeitssystem?

- In welche Einzelaufgaben ist der Gesamtauftrag unterteilt und wie sind diese auf die einzelnen Mitarbeiter verteilt?

Fragen zum technischen Teilsystem:

- Mit welchen technischen Betriebsmitteln und Arbeitsverfahren werden die einzelnen Aufgaben bearbeitet?
- Wo und wie oft treten technisch bedingte Störungen und Schwankungen in Form von Ausfällen, Engpässen usw. auf?
- Welche räumlichen Bedingungen sind gegeben?

Fragen zum sozialen Teilsystem:

- Wie sieht die Mitarbeiterstruktur bezüglich Hierarchie, Ausbildung, Betriebszugehörigkeit, Geschlecht usw. aus?
- Wie sieht die Kooperationsstruktur aus und welchen Zusammenhang hat diese mit dem Auftreten und der Art des Umgangs mit Störungen und Schwankungen im Arbeitsablauf?
- Welche Bedürfnisse bezüglich der Arbeit selbst (Verantwortung, Selbständigkeit, Kooperation und Kommunikation, Weiterentwicklung usw.) und bezüglich der Arbeitsbedingungen (Arbeitszeit, Lohn, Mitsprache usw.) haben die Beschäftigten?

Für die Beantwortung zahlreicher dieser Fragen stellt die Arbeitspsychologie spezielle getestete Instrumente und Verfahren zur Verfügung. Nebst den objektiven Verfahren, mit deren Hilfe Experten z.B. Auftrags- und Bedingungsanalysen, Tätigkeitsanalysen und -bewertungen sowie Anforderungs- und Belastungsanalysen durchführen, erscheinen uns vor allem jene Verfahren relevant, mit deren Hilfe die Beschäftigten ihre Arbeitssituation subjektiv analysieren und bewerten können. Diese Verfahren eignen sich in besonderer Weise für die Mitarbeiterbeteiligung in Veränderungsprozessen. Ihre Ergebnisse können ohne allzu grossen Auswertungsaufwand und gut verständlich zurückgemeldet werden. So bilden sie eine wichtige Grundlage für die Definition von Anforderungen an das neue Arbeitssystem im nächsten Schritt.

4.3.2 Schritt 2: Definition der Anforderungen an das zukünftige Arbeitssystem

Ziel:

Die Ergebnisse der Situationsanalyse und vor allem die Beschäftigung mit den Stärken und Schwächen erlauben es nun, Anforderungen an die zukünftige Situation zu erstellen. Die Anforderungen sollen dabei so formuliert werden, dass daraus konkrete Gestaltungsziele abgeleitet werden können. Anforderungen müssen sich dabei sowohl auf die Bedürfnisse der Organisation wie auch auf die Bedürfnisse der Beschäftigten beziehen.

Methodische Zugänge:

Für die Definition von Anforderungen eignen sich am besten gruppenorientierte Vorgehensweisen, bei welchen Kreativitätstechniken wie Brainstorming oder Metaplan ideal eingesetzt werden können. Bei der Definition von Anforderungen ist es von äusserster Wichtigkeit, dass eventuelle Änderungen am Gesamtauftrag des Arbeitssystems allen Beteiligten klar sind und von ihnen auch unterstützt werden. Zudem ist es gruppendynamisch wichtig, dass das Arbeiten in Gruppen von einem Moderator begleitet wird und dass man sich an vorher gemeinsam aufgestellte 'Spielregeln' hält. Da die Gruppen idealerweise aus Personen mit unterschiedlichen betrieblichen Stellungen zusammengesetzt sind, ist häufig eine Moderation durch einen betriebsexternen Moderator angebracht.

Damit bei der Anforderungsdefinition die ganzheitliche Sicht aus der Situationsanalyse nicht verlorengeht – und dadurch nur Anforderungen zur Lösung isolierter Probleme genannt werden –, lohnt es sich, zu Beginn themenorientiert vorzugehen. Eine mögliche Themengliederung ergibt sich schon aus der Gliederung der Situationsanalyse:

(1) Anforderungen aus den Schwachstellen der Umweltbeziehungen,

(2) Anforderungen aus den Schwachstellen der Abhängigkeiten und dem Arbeitsablauf,

(3) Anforderungen aus den Schwachstellen im technischen Teilsystem,

(4) Anforderungen aus den Schwachstellen im sozialen Teilsystem.

Zu jedem Themenblock liegen aus der Situationsanalyse konkrete Ergebnisse vor, aufgrund deren nun spezifische Anforderungen formuliert werden können. In die Formulierung der Anforderungen sollten jedoch noch keine konkreten Lösungsvorschläge einfliessen.

In einem nächsten Schritt werden die genannten Anforderungen aufeinander bezogen, nach organisations- und beschäftigtenspezifischen Kriterien kategorisiert und – dies ist besonders wichtig – gewichtet. Unterschiedliche Gewichtungen einzelner Anforderungen durch verschiedene Beteiligte weisen auf unterschiedliche Interessenlagen hin. Die Gewichtung ermöglicht es, verschiedene Standpunkte explizit zu machen und Lösungen auszuhandeln.

Anforderungen sollten immer möglichst positiv formuliert werden, z.B.:

Wir brauchen ...
... eine Verringerung des administrativen Aufwandes
... direktere Kommunikationswege
... grössere Aktualität der Informationen
... grössere Nähe zum Kunden

Wir wollen ...
... mehr Verantwortung für die eigenen Aufgaben
... bessere Nutzung der vorhandenen Qualifikationen
... Schaffung neuer Qualifikationsmöglichkeiten
... mehr Abwechslung in der Arbeit

4.3.3 Schritt 3: Konzeption des neuen Arbeitssystems

Ziel:

Bei der Konzeption eines neuen Arbeitssystems werden nun für die in Schritt 2 erstellten Anforderungen konkrete Gestaltungsziele und Lösungen erarbeitet. Die optimale Lösung soll in der anschliessenden Softwareentwicklung als verbindliche Vorgabe für die Gestaltung der technischen Unterstützung dienen.

Methodische Ansätze:

Generell gelten auch bei diesem Schritt jene Moderationsmethoden als geeignet, die viel Platz für Kreativität und Aushandlungsprozesse in Gruppen offenlassen. Da konkrete Gestaltungslösungen immer in mehreren Varianten erstellt werden sollten, lohnt sich deren Erarbeitung in parallelen Arbeitsgruppen. Die einzelnen Lösungsvarianten können dann anhand der Anforderungen aus Schritt 2 evaluiert und gemeinsam zu einer optimalen Variante integriert werden. Damit das Ergebnis der Konzeption des neuen Arbeitssystems auch wirklich als praktikable Leitlinie in der Softwareentwicklung zum Tragen kommt, sollen folgende Fragen beantwortet sein:

(1) Wie sieht die Organisationsstruktur im neuen Arbeitssystem aus?

(2) Wie werden der Arbeitsablauf und die einzelnen Aufgaben gestaltet?

(3) Welches sind die Grundmerkmale der technischen Unterstützung?

Gerade für die Beantwortung der letzten Frage ist die Mitarbeit jener Softwareentwickler, die anschliessend die technische Unterstützung für das neue Arbeitssystem konzipieren und realisieren, unumgänglich. Ihr Beitrag darf jedoch nicht darin bestehen, für die Mitarbeiter gute Lösungsvarianten durch das Anführen technischer Sachzwänge präventiv zu unterbinden. Ihr Auftrag besteht vielmehr darin, im Rahmen der technischen Möglichkeiten mit den zukünftigen Benutzern zusammen jene technische Lösung zu finden, die die gewünschte Form des Arbeitssystems optimal unterstützt.

Zum Schluss dieser Ausführungen wollen wir noch kurz einige Anmerkungen zur Situation bei den Projekttypen C und D (Branchen- und Standardsoftware) anführen. Diese Projekttypen zeichnen sich ja gerade dadurch aus, dass dabei die Software nicht für ein bestimmtes Arbeitssystem entwickelt wird, sondern für eine Klasse von Arbeitssystemen, welche die in der Software enthaltene Funktionalität nutzen können. Auch bei diesen erachten wir es als möglich und notwendig, bei verschiedenen möglichen Kunden Abklärungen zur Arbeitsorganisation vorzunehmen. Zu oft werden diese Systeme nämlich ausschliesslich auf ihre Funktionalität hin entwickelt oder werden aufgrund einer Individuallösung für einen spezifischen Kunden dann in der Branche breit angeboten. Organisatorische Kontexte des einzelnen Anwenders fallen so in der Entwicklung ausser Betracht und zwingen den Anwender oftmals zu ungewollten Anpassungen seiner Arbeitsstrukturen an die Technik. Mit Organisationsanalysen in diversen potentiellen Anwenderbetrieben wird es möglich, prospektiv den Zugang zur Systemfunktionalität derart zu gestalten, dass ein Softwaresystem in unterschiedlichen Organisationsstrukturen flexibel einsetzbar wird.

4.4 VORGEHEN ZUR ERMITTLUNG ERSTER ANFORDERUNGEN AN DIE SOFTWARE

Die Ermittlung der vollständigen und richtigen Anforderungen an eine zukünftige Anwendungssoftware stellt trotz der Fortschritte bei der Werkzeugunterstützung für die Softwareentwicklung weiterhin ein äusserst schwieriges Problemfeld dar.

Es können vier Arten von aufgabenbezogenen Anforderungen unterschieden werden:

(1) Die erste Art bezieht sich auf Anforderungen, die sich aus der Aufgabengestaltung ergeben. Software muss so gestaltet sein, dass sie Möglichkeiten für Aufgabenorientierung in der Arbeit, wie sie im Teil A, Kapitel 1.2 beschrieben ist, schafft oder zumindest nicht beschneidet.

(2) Die zweite Art von Anforderungen ergibt sich aus der Beziehung zwischen Aufgabe, Benutzer und Computer. Sie regeln die Funktionsteilung zwischen Mensch und Maschine so, dass die spezifischen Stärken des jeweiligen Funktionsträgers optimal genutzt werden.

(3) Anforderungen einer dritten Art beziehen sich auf die Benutzungsoberfläche. Sie bestimmen also, wie ein Benutzer auf die implementierte Systemfunktionalität zugreift und wie der Zugriff auf die im System enthaltenen Informationen im Rahmen der Arbeitsaufgabe zu geschehen hat.

(4) Die vierte Art von Anforderungen ergibt sich aus dem Aufgabengegenstand und meint die Vollständigkeit und richtige Beschreibung der zur Aufgabenerfüllung benötigten Daten und deren Beziehungen untereinander.

Betrachtet man die in der heutigen Praxis vorwiegend angewandten Vorgehensweisen und Methoden bei der Anforderungsanalyse und Systemdefinition, so fällt die Konzentration auf die Anforderungen der vierten Art (Daten und deren Beziehungen) auf. Im Hinblick auf die Erarbeitung der technischen Anforderungen für das Funktions- und Datenmodell einer Anwendung erscheint dies auch sinnvoll und notwendig. Geht man davon aus, dass die ersten zwei Arten von Anforderungen (Aufgabenorientierung und Mensch-Computer-Funktionsteilung) bereits bei der Auslegung des Arbeitssystems im Vorfeld des Softwareentwicklungsprozesses umrissen wurden, so wird es im Hinblick auf die Aufgabenangemessenheit einer Software unter dem Aspekt der zunehmenden Anwendungskomplexität immer wichtiger, bereits auch bei der Anforderungsermittlung softwareergonomische Aspekte in der Analysephase mit zu berücksichtigen. Die softwareergonomische Gestaltung darf sich also nicht mehr darauf beschränken, allgemein abgefasste Gestaltungsrichtlinien in der Realisierungsphase ohne Bezug zur Arbeitsaufgabe umzusetzen und anschliessend aufgrund rein evaluativer Verfahren wie Experimente und Benutzerbefragungen zu korrigieren. Aufgabenangemessenheit in der Softwareentwick-

lung heisst, in der Analysephase prospektiv Erkenntnisse zu gewinnen, die Hinweise auf die softwareergonomische Gestaltung im konkreten Aufgabenkontext des Benutzers geben.

Die folgenden Ausführungen zeigen die wichtigsten Aspekte auf, die für die Bearbeitung softwareergonomischer Fragestellungen bei der Anforderungsermittlung im Softwareentwicklungsprozess zu beachten sind. Damit als Ergebnis der Anforderungsermittlung konkrete Gestaltungshinweise für den Systementwurf beim Prototyping gegeben werden können, empfiehlt sich grundsätzlich ein Vorgehen in drei Schritten, die iterativ-zyklisch durchlaufen werden können.

4.4.1 Schritt 1: Analyse der Struktur und Daten von Arbeitsaufgaben

Beim Entwurf des Arbeitssystems als Vorbereitung zum Softwareentwicklungsprozess wurden die einzelnen Aufgaben festgelegt und umschrieben. Damit für die einzelnen Aufgaben nun die optimale technische Unterstützung gefunden weden kann, müssen für jede Aufgabe getrennt die spezifischen Merkmale bezüglich Strukturen und die Datenstruktur des gesamten Systems analysiert werden. Eine besondere Schwierigkeit bietet hierbei oftmals die Tatsache, dass die zu analysierenden Aufgaben in der geplanten Form noch gar nicht bestehen. *Aufgabenstrukturen und Datenstruktur können also nicht direkt beobachtet oder erfragt werden*, sondern sind in einem konstruktiven Prozess zusammen mit den zukünftigen Benutzern und Betroffenen des Systems zu erarbeiten.

Arbeitsaufgaben können anhand sehr unterschiedlicher Merkmale analysiert und beschrieben werden. So stellen etwa Beck und Ziegler (1991) in einem Projekt zur Technik der aufgaben- und benutzerangemessenen Softwarekonstruktion (TASK) ein Modell mit fünfzehn Aufgabenmerkmalen (Name, Art, Struktur, auslösendes Ereignis, Vor- und Nachbedingung, Häufigkeit, Wiederholungsrate, Priorität, Dauer, Vollständigkeit, Eingriffs- und Auswahlmöglichkeiten, Belastungsfaktoren, Restriktionen, technische Durchführbarkeit, Sicherheit, Komplexität) und mit insgesamt über dreissig Beschreibungskategorien dar, mit dem die relevanten Aspekte einer Arbeitsaufgabe für die Softwareentwicklung vollständig zu erfassen sind. Obwohl wir diese Vollständigkeit als richtig und auch notwendig erachten, erscheint uns als Einstieg in die Aufgabenanalyse mit direkter Benutzerbeteiligung eine Konzentration auf die zentralen Merkmale 'Aufgabenstruktur' und 'Variationen in der Abfolge der Teiltätigkeiten' als sinnvoll.

Die Aufgabenstruktur gibt Auskunft über die Teiltätigkeiten, aus welchen eine Aufgabe zusammengesetzt ist, und die Abfolge bzw. Hierarchie der Teiltätigkeiten. Mit den 'Variationen in der Abfolge der Teiltätigkeiten' werden mögliche alternative Abfolgen aufgezeigt, die je nach Arbeitssituation und persönlichem Arbeitsstil genutzt werden können.

Neben der Analyse der Aufgabenstruktur der einzelnen Aufgaben bildet die Analyse der Daten und die Erstellung einer ersten Datenstruktur und -spezifikation für das gesamte Arbeitssystem den zweiten Schwerpunkt des ersten Vorgehensschrittes. Ziel der Datenanalyse ist es, sämtliche für die Bearbeitung der Arbeitsaufgaben benötigten Daten zu erfassen, ihre Merkmalsausprägungen (Wertebereich, Identifikation usw.) so zu beschreiben, dass sie den softwaretechnischen Kriterien genügen und ihre formalen Beziehungen untereinander darstellen. Für die Analyse der Datenstruktur können diejenigen softwaretechnischen Methoden und Vorgehensweisen eingesetzt werden, die der zukünftigen Anwendung und der vorhandenen Entwicklungsumgebung am besten entsprechen.

4.4.2 Schritt 2: Integration der Analyseergebnisse

Ist die Analyse der Aufgabenstruktur der einzelnen Arbeitsaufgaben abgeschlossen und besteht eine erste Sicht der Datenstruktur und -beschreibung, so werden im zweiten Schritt der Anforderungsermittlung die Ergebnisse des ersten Schrittes derart integriert, dass jeder Aufgabe die dazugehörenden Daten mit ihren spezifischen Merkmalsausprägungen zugeordnet werden. Als spezifische Merkmalsausprägungen gelten zum Beispiel bestimmte Werte aus dem gesamten Wertebereich, die für diese bestimmte Aufgabe zugelassen sind. Mit diesem Vorgehen kann nun festgelegt werden, an welchen Stellen im gesamten Aufgabenkontext ein Datum in welcher Form gebraucht wird, welchen aufgabenspezifischen Bedingungen es jeweils genügen muss, wo es erfasst und geändert werden kann oder wo es reinen Informationszwecken dient.

4.4.3 Schritt 3: Ableitung softwareergonomischer Anforderungen

Damit aufgrund der vorliegenden Informationen zum Zusammenhang zwischen Aufgabenstruktur und Daten erste Anforderungen für den anschliessenden Entwurf abgeleitet werden können, müssen die einzelnen Teiltätigkeiten der Aufgaben noch bezüglich ihres Schwierigkeitsgrades oder nach der zur Bewältigung der Aufgaben erforderlichen kognitiven Anforderung

klassifiziert werden. Ein mögliches Klassifikationsschema für kognitive Anforderungen ist etwa die Unterscheidung der Teiltätigkeiten in solche der reinen Informationsübertragung (die kognitive Anforderung beschränkt sich aufs Wahrnehmen und kurzzeitige Behalten von Informationen), der Informationsverarbeitung (Informationen werden nach vorgegebenen Regeln zugeordnet oder ausgewählt), der Informationsbearbeitung (aus bestehenden Informationen werden nach vorgegebenen Regeln durch Beurteilen, Ordnen und Verknüpfen neue Informationen geschaffen) oder der Informationserarbeitung (neue Informationen werden ohne vorgegebene Regeln erarbeitet). Diese Aufzählung von Klassifikationsmöglichkeiten macht deutlich, dass innerhalb einer einzelnen Aufgabe Teiltätigkeiten mit verschiedenen Anforderungshöhen auftreten können. Mit der Differenzierung innerhalb einer Aufgabe wird es nun möglich, für die einzelnen Teilschritte bereits im Vorfeld zum Prototyping gewisse Gestaltungsregeln der Softwareergonomie als Anforderungen an das technische System festzuhalten, also z.B. zu bestimmen, ob eine bestimmte Teiltätigkeit besser mit einer Informationsdarstellung in Listenform oder in grafischer Form dargestellt wird. Mit der Benennung solcher Anforderungen vor dem Prototyping können auch gezielt Evaluationskriterien aufgestellt werden, die bei der Überprüfung von Prototypen als Orientierungsrahmen verwendet werden können.

Wir werden im folgenden die meisten bekannten Methoden einzeln darstellen, mit denen sich Benutzer an verschiedenen Stellen des Entwicklungsprozesses beteiligen lassen.

4.5 DER WORKSHOP

Neben mündlichen und schriftlichen Umfragen sind Workshops wohl die am meisten genutzte Möglichkeit zum Einbezug von Benutzern; diese Methode eignet sich sehr gut für die Zusammenarbeit von Entwicklern und Benutzern in verschiedenen Stadien des Entwicklungsprozesses, sei es zur Analyse grundlegender Probleme, zur Bewertung verschiedener Lösungsvarianten oder zur Gestaltung der Benutzungsoberfläche. Unter der Bezeichnung 'Joint Application Design' hat sich in Nordamerika in manchen Firmen die *gezielt in den Entwicklungsprozess eingebaute Durchführung mehrerer, stark strukturierter Benutzer-Entwickler-Workshops eingebürgert* (siehe dazu Wood und Silver 1989 sowie August 1991).

Workshops sind eine ausgezeichnete Möglichkeit für die *direkte* Zusammenarbeit von Entwicklern und Benutzern und nutzen die Vorteile der

Gruppendynamik beim Problemlösen. Sofern Workshops nicht schon standardmässig im Entwicklungsprozess vorgesehen sind, können unterschiedliche Ereignisse den Anlass für die Durchführung eines Workshops bilden: z.B. Ergebnisse einer Benutzerumfrage, ein Meilenstein im Projekt, der Wunsch des Projektleiters, offene Fragen mit Benutzern direkt zu klären, usw. Für einen produktiven, erfolgreichen und für alle Beteiligten gewinnbringenden Workshop ist eine gute *Vorbereitung*, eine *Aufbereitung* der Ergebnisse sowie oftmals eine *Moderatorenschulung notwendig.*

4.5.1 Vorbereitung

Es ist unbedingt erforderlich, ein Konzept des Workshops zu erarbeiten, das heisst, Thema, Ziele, Ablauf, Zeitplan, Teilnehmer, Moderation, Fragen, Aufgaben, Methoden, Material, Geräte (z.B. Prototypingwerkzeuge), Räume, Schulungsnotwendigkeiten (z.B. für bestimmte Kreativitätstechniken) zu klären! Für eine detailliertere Darstellung siehe Wood und Silver (1989) sowie August (1991).

Als Teilnehmer sollten für den Anwendungsbereich repräsentative, qualifizierte und interessierte Benutzer, Mitglieder der Projektgruppe (Entwickler) sowie evtl. Spezialisten für bestimmte Fragen (z.B. Werkzeuge, Ergonomie) ausgewählt werden. Die Anzahl der Teilnehmer sollte zehn bis zwölf Personen nicht übersteigen; bei grösserer Teilnehmerzahl ist die gezielte Bildung von Arbeitsgruppen – mit evtl. unterschiedlichen Aufgaben – empfehlenswert. Die Ergebnisse können dann im Plenum zusammengetragen werden. Eine gute Vorbereitung der Teilnehmer durch intensives Studium im voraus abgegebener Unterlagen verkürzt die Einstiegszeit!

4.5.2 Ablauf

Ein möglicher grober *Ablauf* eines Workshops sieht wie folgt aus:

(1) 'Spielregeln' erläutern (siehe Checkliste 2 im Anhang).

(2) Einleitung: Ziel und Zweck; eventuell Präsentation von Umfrageergebnissen, Stand der Dinge.

(3) Rahmenbedingungen abklären.

(4) Prioritäten setzen.

(5) Input (Thema umreissen, erste Frage) durch Moderator:
    • Diskussion, Visualisierung;

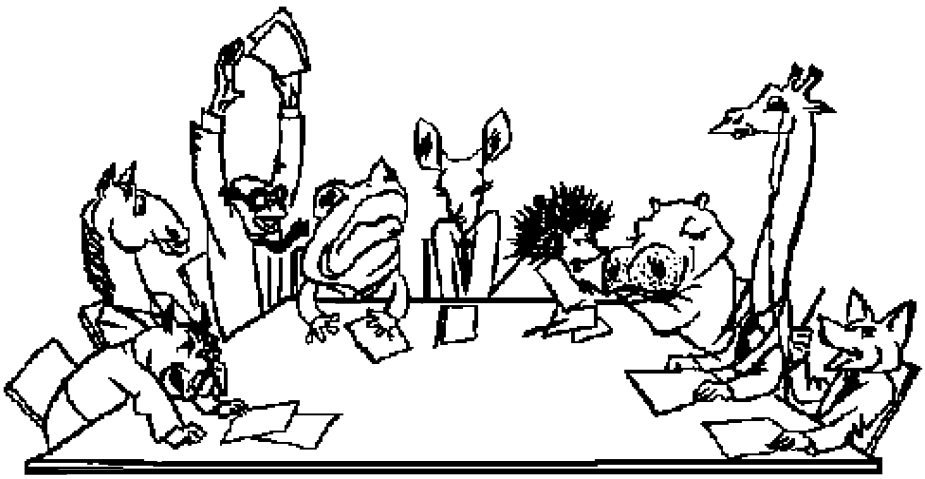
- Zusammenfassen, Visualisierung, Standort bestimmen;
- Diskussion, Visualisierung.

(6) Zusammenfassung, Visualisierung der Ergebnisse, evtl. Ergebnispräsentation von Arbeitsgruppen.

(7) Konsequenzen; Umsetzung; weiteres Vorgehen, Aufgabenverteilung.

Ein *gemeinsames* Mittagessen fördert Kontakte und erlaubt Gespräche in einem informellen Rahmen!

Der Diskussionsverlauf sollte vom Moderator optisch aufgezeichnet werden. Diese *Visualisierung* (z.B. mittels Flipchart, Metaplan nach Mauch 1981, Overhead, Pinwand usw.) dient als 'roter Faden' der Diskussion; Wiederholungen werden vermieden, Redundanzen aufgedeckt, kein Gedanke geht verloren, und Komplexität kann reduziert werden. Ausserdem entsteht dadurch gleichzeitig ein grafisches Protokoll. Die Dokumentation kann durch Protokollführung, Tonbandaufnahmen, Fotos und Videoaufnahmen ergänzt werden.



Der Streiter	**Der Positive**	**Der Alles-wisser**	**Der Red-selige**	**Der Schüch-terne**	**Der Ablehnende**	**Der Un-interes-sierte**	**Das grosse Tier**	**Der Aus-frager**
Sachlich und ruhig bleiben Die Gruppe veran-lassen, seine Be-haup-tungen zu wider-legen	Ergeb-nisse zu-sammen-fassen lassen Bewusst in die Gruppe einschal-ten	Die Gruppe auf-fordern, zu seinen Behaup-tungen Stellung zu nehmen	Takt-voll unter-brechen Rede-zeit fest-legen	Einfache direkte Fragen stellen Sein Selbst-bewusst-sein stärken	Seine Kennt-nisse und Er-fah-rungen aner-kennen	Nach seiner Arbeit fragen Bei-spiele aus seinem Inter-essenge-biet geben	Keine direkte Kritik üben "Ja-aber"-Technik	Seine Fragen an die Gruppe zurück-geben

Abbildung 14: Verhaltensstile sowie darauf abgestimmte Verhaltensweisen des Moderators.

4.5.3 Moderatorenschulung

Erfahrungen bisheriger Workshops haben die *grosse Bedeutung der Rolle des Workshopleiters bzw. Moderators und seiner Fähigkeiten, speziell seiner sozialen Fertigkeiten,* für das Gelingen des Workshops gezeigt. Softwareentwickler und Benutzer haben aber in der Regel keine Ausbildung hinsichtlich der Anleitung von Arbeitsgruppen bzw. der Leitung von Diskussionsgruppen erhalten und fühlen sich in derartigen Situationen vielfach auch unsicher. Ausserdem neigen Entwickler auch häufig zur Verteidigung ihrer Produkte und wehren Benutzervorschläge mit technischen Argu-

menten ab, was alles den positiven Ablauf eines Workshops und die Zielerreichung gefährden kann. Manchmal ist der Einsatz einer 'neutralen' Person (weder Benutzer noch Entwickler, evtl. Aussenstehender) als Moderator von Vorteil! Eine Moderatorenschulung sollte folgende Aspekte zumindest rudimentär behandeln. Die Moderatoren werden sich dann sicherer fühlen, und der Workshop wird problemloser und zufriedenstellender verlaufen!

- Verständnis der Moderatorenrolle:
  Ein Moderator ist weder ein Lehrer, noch ein Referent, noch ein Verkäufer eigener Ideen, sondern ein 'Vermittler', seine Hauptaufgabe besteht in der Strukturierung der Situation und des Ablaufs im Sinne der Zielerreichung!
- Gesprächsführung:
  Grundsätzlich *nichtdirektiv*; keine Verteidigung, keine Abwehr a priori (z.B. "geht nicht"); nicht eigene Hypothesen bestätigen oder durchsetzen wollen; aber: erklären (z.B. technische Probleme), realistische Erwartungen wecken.
- Diskussionsleitung:
  In der Diskussion anregen, steuern, ausgleichen, erläutern, fragen, rückfragen, hinterfragen, Bezüge herstellen, zusammenfassen, Details nicht ins Zentrum rücken, Konzentration auf das Wesentliche! Umgang mit verschiedenen 'Diskussionstypen' (siehe Abbildung 14).

4.5.4 Aufbereitung – Feedback – Umsetzung

Nach dem Workshop müssen die Ergebnisse *aufbereitet* und – zumindest in zusammengefasster Form – an die Teilnehmer *zurückgemeldet* werden. Ausserdem erreicht man eine enorme Motivationswirkung für die weitere Zusammenarbeit mit Benutzern, wenn z.B. kurzfristig mit geringem Aufwand realisierbare Vorschläge zur Systemverbesserung *umgesetzt* werden; die Benutzer erleben dadurch, dass ihre Beiträge ernst genommen werden und sie wirklich einen Einfluss auf die Systementwicklung haben!

4.6 BEFRAGUNG

4.6.1 Schriftliche und mündliche Befragung

Befragungen lassen sich wie folgt unterscheiden:

(1) Mündliche Befragung in Form des Interviews (siehe Richardson, Dohrenwend und Klein 1965; Stewart 1983, James 1989, Froschauer und Lueger 1992),

(2) schriftliche Befragung mit Hilfe von Fragebogen (siehe auch Mummendey 1987),

(3) schriftliche Befragung mit Hilfe eines 'interaktiven', elektronischen Fragebogens (z.B. per 'electronic-mail').

Folgende Aspekte sind bei einer Befragung zu berücksichtigen.

Strukturiertheitsgrad:

Je weniger man über ein Gebiet weiss, je weniger präzise Vorstellungen oder Hypothesen existieren, desto geringer strukturiert muss die Befragung sein – und umgekehrt. Eine grobe Unterscheidung kann getroffen werden zwischen

(1) nichtstandardisierter Befragung,

(2) teilstandardisierter Befragung und

(3) standardisierter Befragung.

Typen von Fragen:

In Abhängigkeit vom Strukturiertheitsgrad können die einzelnen Fragen im Interview bzw. im Fragebogen wie folgt unterschieden werden. Es gibt:

(1) *Offene Fragen*, das heisst, Fragen ohne vorgegebene Antwortmöglichkeiten,

(2) *geschlossene Fragen* mit unterschiedlichen Antworttypen (Alternativfragen [z.B. 'ja' / 'nein'], Mehrfachwahlfragen, Skalafragen usw.),

(3) *direkte Fragen* (Sinn und Zweck der Frage ist erkennbar),

(4) *indirekte Fragen* (Sinn und Zweck der Frage ist verschleiert, um bewusst verfälschte Antworten zu vermeiden).

Formulierungsregeln:

Der Erfolg einer Befragung hängt u.a. von der Qualität der Fragen ab. Nachfolgend sind einige Regeln aufgeführt, die beachtet werden sollten:

(1) Einfache Formulierungen wählen,

(2) lange Fragen vermeiden,

(3) Suggestivfragen vermeiden,

(4) nur eindeutig beantwortbare Fragen stellen (z.B. keine Doppelfragen),

(5) doppelte Verneinung vermeiden,

(6) Überforderung des Antworters durch zu umfangreiches Material vermeiden,

(7) konkrete Fragen sind besser als allgemeine.

Erhebungen in Form von *schriftlichen Befragungen* haben gegenüber persönlichen und telefonischen Interviews folgende Vorteile:

(1) Die befragte Person kann den Fragebogen dann ausfüllen, wenn sie Zeit dazu hat, allenfalls auch in mehreren Schritten zu verschiedenen Zeitpunkten.

(2) Die Situation bei der Beantwortung ist gegebenenfalls vollständig anonym und vertraulich.

(3) Erläuternde Unterlagen können, auch wenn sie komplex sind, ohne Zeitdruck durchgelesen werden.

(4) Nicht zuletzt entfällt der Kontaktierungs- und Befragungsaufwand beim Erheber, was zu einem kostengünstigen Vorgehen führt.

Wichtig bleibt festzuhalten, dass bei betrieblichen Befragungen die Befragten *während* der Arbeitszeit ausreichend Gelegenheit zum Ausfüllen der Fragebogen erhalten.

Zu den wesentlichen Nachteilen der schriftlichen Befragung gehört, dass über Gründe und Motive derjenigen, die nicht antworten, keinerlei Informationen vorliegen. Erschwerend kommt bei schriftlichen Befragungen hinzu, dass die Rücklaufquoten – je nach Zielpublikum, Gültigkeit und Anzahl von Adressen, Streuung des Fragebogens und Administration des Mahnwesens – stark unterschiedlich sein können; im allgemeinen sind Rücklaufquoten unter 30% als kritisch, 30–50% als akzeptabel und über 50% als gut bis sehr gut anzusehen. Niedrige Rücklaufquoten können dann vorkommen, wenn der Fragebogen wenig ansprechend gestaltet ist, eine grosse Zahl von Fragebogen an ein breitgefächertes Zielpublikum verschickt wird (z.B. bei Umfragen in Zeitschriften) oder die Adressen nicht mehr auf dem aktuellen Stand sind. Dagegen können bei regelmässigen telefonischen Befragungen über 60% des Zielpublikums erreicht und inter-

viewt werden. Eine ausführliche Einführung in das Gebiet der schriftlichen Befragung geben Mummendey (1987) und Vögele (1992).

Bei einer *geringen Rücklaufquote* sind die Ursachen hierfür ausfindig zu machen. Der beste und in der Praxis auch am häufigsten beschrittene Weg, um hier mehr Informationen erhalten zu können, besteht in einer direkten telefonischen Nachfrage bei einer repräsentativen (z.B. zufällig ausgewählten) Stichprobe aller Nichtantwortenden. Da nur das Motiv der Nichtteilnahme sowie einige grundlegende zusätzliche Angaben interessieren, kann das Telefongespräch kurzgehalten werden; die Kosten für eine derartige Nacherhebung sind daher eher gering.

Für eine sauber durchgeführte Nacherhebung bedarf es jedoch einer gut organisierten Rücklaufadministration, bei der eindeutig festgestellt werden kann, wer nicht – bzw. noch nicht – geantwortet hat. Damit ist keineswegs die Vertraulichkeit der bei der postalischen Befragung gemachten Angaben in Frage gestellt. Entscheidend ist dabei, dass der Auftraggeber der Umfrage keine Rückschlüsse auf einzelne Antwortende ziehen kann. Bei einer solchen Nachbefragung von 300 Nichtteilnehmern aus einer Umfrage unter 5000 Personen ergaben sich die folgenden Begründungen für eine Nichtteilnahme: 40% hatten 'keine Zeit', 30% gaben an, 'den Fragebogen nie gesehen zu haben', 10% fühlten sich 'nicht zuständig', für 5% war 'der Fragebogen zu umfangreich', 20% hatten andere Begründungen.

4.6.2 Befragung in elektronischer Form

Befragungen auf elektronischem Wege (email) sind – abgesehen von der Programmerstellung des 'interaktiven' Fragebogens – gegenüber schriftlichen Befragungen nahezu kostenlos. Dafür können jedoch grundsätzlich nur Personen erreicht werden, welche über einen entsprechenden elektronischen Anschluss verfügen.

Beim Einsatz von 'interaktiven' Fragebogen in elektronischer Form sind folgende positiven und negativen Aspekte zu berücksichtigen:

- Es gibt keine Kontrolle darüber, ob eine Person mehrmals antwortet.
- Nur Benutzer mit Zugang zu einem Computer mit email-Anschluss können befragt werden.
- Es sind im Netzwerk und im 'interaktiven' Fragebogen besondere softwaretechnische Sicherheitsvorkehrungen gegen Eindringlinge von 'aussen' zu treffen.

– Da ein elektronischer Fragebogen kein normaler Bogen aus Papier ist, der geduldig auf dem Schreibtisch auf seine Beantwortung wartet, bedarf es zusätzlicher Motivatoren, um die Bearbeitung in den Aufmerksamkeitsfokus des Benutzers zu rücken.

+/- Es bedarf einer guten Informationspolitik im organisatorischen Vorfeld.

+/- Frageschleifen mit stets ähnlichen Fragen sollten vermieden werden ('keep it simple').

+ Programmierbare Verzweigungen lassen den Fragenverlauf in Abhängigkeit von den konkreten Antworten von überflüssigen Fragen freihalten.

+ Die Abspeicherung der Antworten kann gleich in einem für ein Statistikauswertungsprogramm passenden Format erfolgen.

+ Bei regelmässigem Einsatz sind die Kosten für die Datenerhebung und -auswertung verschwindend gering.

+ Es kann ein beliebig grosser Benutzerkreis angesprochen werden.

4.7 DAS PFLICHTENHEFT

Das Pflichtenheft umfasst in der Regel alle Anforderungen an das zu erstellende System (Hard- und Software); die Anforderungen umfassen die funktional-technischen, die hard- und softwareergonomischen sowie die wirtschaftlichen Aspekte (siehe Schreiber 1994). Das Pflichtenheft ist in der Regel als Textdokument gegeben. Um jedoch unter allen Betroffenen ein möglichst gemeinsames Verständnis zu ermöglichen, empfiehlt es sich, die Inhalte des Pflichtenheftes zusätzlich auch in anschaulicher Form darzustellen. Bevor man jedoch mit formalen Beschreibungssprachen arbeitet, sind einfachere und anschaulichere Verfahren angesagt: z.B. Simulationsmethoden und Prototyping.

4.8 SIMULATIONSMETHODEN

Unter Simulationsmethoden werden hier alle Verfahren zusammengefasst, welche Attrappen verschiedenster Art verwenden und für ihre Erstellung und Anwendung möglichst wenig Aufwand erfordern (je einfacher und preiswerter, desto besser! Siehe auch Ehn und Kyng 1991). Um diesem Anspruch in den jeweiligen Arbeitssitzungen, Workshops, Benutzertreffen usw. nachkommen zu können, wird von allen Beteiligten ein ausreichendes Mass an Offenheit, Neugierde und Kreativität benötigt.

Kasten 19: Ein Beispiel für eine 'Wizard of oz'-Simluationsstudie.

In einem grossen europäischen Forschungsprojekt aus dem Jahre 1990 zur Entwicklung eines Datenbankabfragesystems mit einer natürlichsprachlichen Ein- und Ausgabe wurde folgende Simulationsstudie ('wizard of oz') durchgeführt: Ein normales Bildschirmgerät im Lesesaal einer Bibliothek, welches ursprünglich für die Literaturrecherche mittels einer traditionellen Bibliothekssoftware mit Menüoberfläche eingesetzt wurde, bekam eine 'natürlichsprachliche' Benutzungsoberfläche; dies wurde den Benutzern auf entsprechenden Hinweistafeln mitgeteilt. In der Tat wurde das Bildschirmgerät jedoch lediglich mit einem anderen Bildschirmgerät im Raum eines Bibliotheksangestellten nebenan verbunden, welcher dann die Eingaben der Benutzer las und über ein zweites, traditionelles Gerät die entsprechenden Recherchen durchführte. Die Ergebnisse wurden dann vom Angestellten 'von Hand' auf dem simulierten Bildschirmgerät ausgegeben.
Die Ergebnisse waren verblüffend: Die natürlichsprachlichen Eingaben der Benutzer waren oft grammatikalisch falsch, mit einer schlechten Satzstellung und weitgehend dürftig formuliert; es traten sehr viele Buchstabierfehler auf, dagegen erstaunlich wenig Fragesätze; ein hoher Prozentsatz der Sätze enthielt kein Verb oder bestand sogar nur aus komplexen Hauptwortkonstruktionen; die Benutzung von Fürwörtern war auf einige wenige beschränkt. Einige Benutzer fingen sogar an, dem System Fragen zu politischen Problemen des Weltgeschehens (z.B. "Was macht Gorbatschow gerade?") – bzw. anderen nicht bibliotheksbezogenen Problemen – zu stellen.

Es muss nicht immer die Hochglanzfolie sein, wenn es schon eine einfache Handskizze tut. Am besten bewähren sich Medien (z.B. [Wand-]Tafeln), auf denen man gemeinsam zeichnen und beliebig verbessern kann. Für die Gestaltung grafischer Oberflächen gibt es bereits spezielle Methoden (z.B. PICTIVE von Muller 1993). Das wichtigste ist jedoch, dass die Beteiligten anfangen, gemeinsam etwas zu analysieren, zu bewerten und zu verbessern.

Im Kasten 19 ist eine Simulationsstudie beschrieben, welche durchgeführt wurde, um das zu erwartende Benutzerverhalten bei der geplanten System-

version im vorhinein studieren zu können. Es zeigten sich nicht vorhergesagte Erwartungshaltungen seitens der Benutzer.

4.8.1 Szenarium

Szenarien sind Situationsbeschreibungen über relevante, gegenwärtige oder angestrebte Arbeitssituationen und umfassen: auszuführende Handlungen, eingesetzte oder einzusetzende Arbeitsmittel und die Motivation bzw. Gründe für die einzelnen Handlungen, Strukturen usw. Ein Szenario ist durch die folgenden Merkmale gekennzeichnet:

(1) Es geht um *einen einzelnen* realen – oder fiktiven – Benutzer.

(2) Von diesem Benutzer wird im Rahmen seiner Aufgabenbearbeitung eine bestimmte Anzahl an modernen Werkzeugen eingesetzt.

(3) Es soll bei der beschriebenen Aufgabenbearbeitung ein ganz bestimmtes Arbeitsergebnis erzielt werden.

(4) Dies alles spielt sich in einer konkret beschriebenen Arbeitsumgebung unter genau angegebenen Bedingungen ab.

Szenarien werden in der Regel von ein bis zwei Personen erstellt, welche ganztägig für zirka zwei Wochen daran arbeiten. Dazu gehören: Auswahl eines Problem- bzw. Arbeitsbereiches; Befragung der Fachanwender oder anderer relevanter Ansprechpartner; Brainstorming über die Verwendung und den möglichen Einsatz neuer Technologien; Beschreiben des Szenarios auf ein bis zwei Seiten in einem erzählenden Stil, welcher möglichst lebendig, realistisch und gut nachvollziehbar wirkt; technische und andere Details können im Anhang zum Szenario dargestellt werden. Jeder, der das Szenarium gelesen hat, sollte anschliessend mit eigenen Worten erklären können, worum es dabei geht und was die wesentlichen neuen Ideen und Aspekte sind (siehe auch bei Meyer-Schönherr 1992).

Da bei einem einzelnen Szenario die Gefahr besteht, dass es zu stark auf bestimmte Vorstellungen fokussiert und damit in seiner Aussagekraft eingeschränkt ist, empfiehlt es sich, mehrere Szenarien anzufertigen. Entwickler haben dann durch solche Szenarien die Möglichkeit, sich einen detaillierten Einblick in zukünftige Anwendungen der Benutzer zu verschaffen.

4.8.2 Drehbuch

Während Szenarien sich eher auf das gesamte Arbeitsumfeld beziehen, lassen sich *Drehbücher* ('storyboards'; siehe Andriole 1989) als Simulations-

methoden für einzelne Abläufe einsetzen. Nicht wie im Film – obwohl der Begriff von dort übernommen wurde – sind Drehbücher ein Textdokument, sondern vielmehr eine Abfolge von Bildern (z.B. 'mock-ups', Handskizzen usw.). Hat man z.B. einen Grafikeditor auf einem PC zur Hand und zusätzlich noch ein Datenbankprogramm, welches Bilder verwalten kann, so lässt sich mit verhältnismässig geringem Aufwand ein ganzer Komplex von Abläufen erzeugen und nach verschiedenen Kriterien zusammenstellen. Selbst wenn man diese Werkzeuge nicht hat, so kann man mit ganz einfachen Papier- und Bleistiftskizzen arbeiten, welche kopiert, verändert, überarbeitet und durchnumeriert werden. Werden Szenarien mit Drehbüchern gekoppelt, so kann daraus eine *Videodemonstration* entwickelt werden. Bei innovativen Problemstellungen, für die es noch keine vergleichbaren Lösungen gibt, ist eine echte *Simulationsstudie* oftmals die einzige Möglichkeit, um das Benutzerverhalten im vorhinein studieren zu können (siehe Kasten 19).

4.9 FORMALE DARSTELLUNGSMETHODEN

Bei informations- und aktivitätsorientierten Darstellungsmethoden aus der Informatik wird in formalisierter Form die Veranschaulichung des geplanten Systems verfolgt. Dafür werden gleichermassen Entscheidungstabellen, Konfigurations-, Funktions-, Aktivitäten-, Datenflussablauf- und Programmablaufdiagramme wie z.B. HIPO, JACKSON, YOURDAN, Petri-Netze, SADT usw. eingesetzt (speziell für interaktive Systeme siehe Dix 1991). Allgemein zeichnen sich die Analyse- und Darstellungsmethoden aus der Informatik durch eine zum Teil sehr komplexe Syntax und abstrakte Aktivitäteneinteilung aus und sind in diesem Sinne meist nicht als 'beteiligungsorientiert' zu bewerten. Sie eignen sich am ehesten für die Zusammenarbeit von EDV-Fachleuten (siehe Martin und McClure 1985 sowie Raasch 1991) und sind für eine saubere Spezifikation oft unentbehrlich.

Der Modellansatz von Oberquelle (1987) liefert nun einen Beitrag zur Entwicklung einer *benutzerorientierten Beschreibungssprache* für Organisationen mit interaktiven Computeranwendungen. Dieser Ansatz ermöglicht die Darstellung der abstrakten Gesamtarchitektur einschliesslich der Benutzungsmodelle von computergestützten Teilen aus der Sicht der Benutzer, um den Softwareentwicklungsprozess in seiner Gesamtheit unterstützen zu können. Es werden unter anderem Rollen-Funktions-Aktions-Netze (RFA-Netze) eingeführt, welche Arbeitsorganisationen mit interaktivem Rechnereinsatz beschreiben können.

4.10 PROTOTYPING

Während in der Produktgüterindustrie ein Prototyp – von Designstudien abgesehen – in der Regel ein voll funktionstüchtiges und fehlerfreies Produkt darstellt, ist ein Prototyp im Bereich Software ein irgendwie unfertiges Produkt: Sei es, dass die eigentliche Funktionalität oder die eigentliche Oberfläche fehlt, sei es, dass das Antwortzeitverhalten unzureichend ist, oder sei es, dass die interne Programmstruktur nicht nach gängigen softwaretechnischen Kriterien ausgestaltet ist (siehe Pomberger und Blaschek 1993). Jedenfalls sollte ein Softwareprototyp *operational* sein; ob diese Operationalität direkt gegeben sein muss oder auch simuliert werden kann, wird unterschiedlich beurteilt. Softwareprototyen sollten im Rahmen einer Prototypingstrategie *schnell* und *kostengünstig realisiert* werden können.

Prototyping ist im Kontext partizipativer Softwareentwicklung eine Spezifikationsmethode, bei der die softwareergonomischen Anforderungen zur Benutzungsoberfläche präzise festgelegt werden können (siehe Floyd 1984). Darüber hinaus kann ein Prototyp zur Erprobung und Evaluation von Lösungsvorschlägen, zur Beurteilung der Qualität des Entwurfes einer Systemarchitektur und zur Prüfung verschiedener Realisierungsmöglichkeiten von Systemkomponenten eingesetzt werden. Im Unterschied zu einem (nur simulierten) Softwaremodell ist ein Prototyp immer ein Stück betriebsfähige Software.

4.10.1 Horizontales und vertikales Prototyping

Während ein *horizontaler Prototyp* eine weitgehend funktionslose Fassade ('mock up') darstellt, umfasst ein *vertikaler Prototyp* nur das rein funktionale Modell ohne die angestrebte Benutzungsoberfläche (siehe Abbildung 15). Horizontale Prototypen eignen sich hervorragend für Benutzer-Entwickler-Diskussionen. Vertikale Prototypen dagegen eignen sich eher zum Austesten von Anwendungsfunktionen im Rahmen von Machbarkeitsstudien. Wenn ein zunächst unvollständiger Prototyp zum vollständigen Zielsystem ausgebaut wird, spricht man von einem Pilotsystem *(evolutionäre Prototypingstrategie).* Mit immer mächtiger werdenden Entwicklungsumgebungen, insbesondere bei objektorientierter Programmierung – ggf. zusammen mit einer objektorientierten Spezifikationsmethode –, kann Prototyping fliessend in die eigentliche Systementwicklung übergehen.

Prototyping als eine benutzernahe Methode für den Softwareentwicklungsprozess hat inzwischen eine breite Verwendung gefunden (Kieback et al. 1991) und wird von vielen Unternehmen schon regelmässig eingesetzt.

Prototyping wird dabei hauptsächlich zur methodischen Unterstützung von Konzeptions- und Spezifikationsaufgaben (Quadrant-II) herangezogen und dient vorwiegend – beim horizontalen Prototyping – der Abklärung von Fragen zur Benutzungsoberfläche. Insgesamt wird das Vorgehen mit Prototyping von den Unternehmen als *sehr nützlich* bewertet, da es eine sinnvolle Problemlösemethode für den Softwareentwicklungsprozess darstellt und dabei vor allem auch die Zusammenarbeit mit den Benutzern erleichtern kann. Mit dem Einsatz von Prototyping gehen signifikant geringere Kosten- und Terminüberschreitungen einher! (vgl. Rauterberg und Strohm 1992).

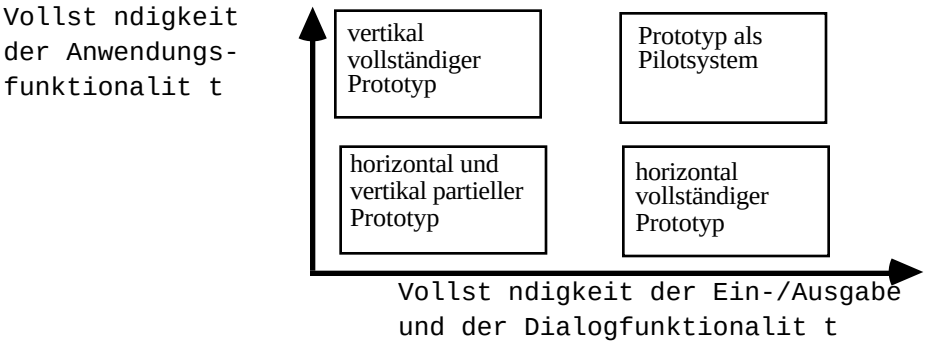


Abbildung 15: Übersicht über die verschiedenen Arten von Softwareprototypen.

Eine Voraussetzung für den Einsatz des Prototyping bilden Softwarewerkzeuge, die eine schnelle und ökonomische Produktion von Prototypen erlauben. In diesem Zusammenhang sind Masken- bzw. Reportgeneratoren, Dialogsystem-, Programm- und Datengeneratoren als wichtige Voraussetzungen für die effiziente Gestaltung eines iterativ-zyklischen Entwicklungsprozesses zu nennen. Endbenutzerorientierte Werkzeuge, die die Produktion von Systembeispielen durch den Benutzer selbst ermöglichen, gewinnen immer stärker an Bedeutung. Im Rahmen von Sprachen der vierten Generation werden vermehrt integrierte Entwicklungssysteme vorgestellt, die gleichermassen das Projektmanagement, die Analysen, den Entwurf und die Systemrealisierung unterstützen sollen.

Erfahrungen bei der Verwendung von Prototypen zeigen, dass bei der Evaluation von Entwurfsvarianten zusammen mit Benutzern eine Menge an

neuartigen Designvorschlägen für den Softwareentwickler anfällt, allerdings auf eine eher unsystematische Weise und durchaus nicht immer nur auf die eigentlich diskutierten Probleme bezogen. Viele dieser Hinweise beziehen sich auf Aspekte, die mit dem Prototypingwerkzeug z.T. gar nicht modelliert werden können. Darunter fallen u.a. Spezifikationen der Funktionalität einzelner Systembefehle. Diese Hinweise müssen dann extra schriftlich festgehalten werden und können erst später (z.B. bei der Programmierung des Systems) berücksichtigt werden. Ein wesentlicher Vorteil des Prototyping besteht darin, dass die Kommunikation zwischen Benutzern, Analytikern und Entwicklern auf eine für alle Beteiligte anschauliche Art angeregt und aufrechterhalten wird.

4.10.2 Probleme beim Prototyping

Bei der Evaluation von Prototypen unter Einbezug der Benutzer stellt sich die Frage nach der Gewichtung und Interpretation der Ergebnisse. Dabei treten verschiedene Probleme auf:

(1) Die Ergebnisse sind kaum quantifizierbar; es lässt sich nicht immer feststellen, ob eine Entwurfsvariante besser ist als eine andere.

(2) Die Aussagen verschiedener Benutzer können widersprüchlich sein – insbesondere über mehrere Evaluationszyklen hinweg.

(3) Es kann durchaus vorkommen, dass die Benutzer bei der Evaluation von Entwürfen in bezug auf die Aufgabengestaltung gegen ihre eigenen Interessen entscheiden, indem sie ihre eigenen Aufgaben möglichst vereinfachen und ihre eigenen Handlungsspielräume einschränken. Sie handeln sich damit für ihre spätere Arbeit Nachteile ein.

Das *erste Problem* tritt beim Bewerten mehrerer alternativer Entwürfe auf, wenn eine beste Variante bestimmt werden soll. Ziel des Entwurfs und der Evaluation zusammen mit den Benutzern ist aber nicht primär die endgültige Bewertung von Entwürfen, sondern das Erarbeiten einer *guten* Lösung. Jeder Entwurf hat seine Stärken und Schwächen, eine brauchbare Lösung lässt sich häufig durch eine Kombination einzelner Elemente der verschiedenen Entwürfe zusammenstellen.

Ein *zweites Problem* sind die widersprüchlichen Aussagen und Veränderungsvorschläge der Benutzer. Sie sind häufig ein Indikator dafür, dass der zu beurteilende Entwurf von einer zu formalisierten Auffassung der Arbeitsaufgabe ausgeht und die Freiheitsgrade der 'widersprüchlichen' Benutzer unnötig einschränkt. Es sollte in diesem Fall nach einer Lösung gesucht

werden, welche durch eine weitergehende Differenzierung genügend flexibel und individuell anpassbar ist, um die möglicherweise widersprüchlichen Vorschläge verschiedener Benutzer zu vereinen.

Der *dritte Problembereich* dagegen ist ein methodisches Problem, welches bei der Evaluation von Prototypen ohne die Beteiligung von Arbeitsgestaltern auftreten kann. Da die in der Regel nicht funktionstüchtigen Prototypen von den Benutzern nur über einen kurzen Zeitraum hinweg evaluiert werden, stehen primär Fragen der Erlernbarkeit im Vordergrund. Die Benutzer – insbesondere Computerneulinge – werden Entwürfe vorziehen, welche eine starke Führung durch das System aufweisen. Solche Systeme erweisen sich aber nach einigem Gebrauch (Tage bis Wochen) als zu inflexibel und werden dann von den Benutzern – insbesondere den an der Entwicklung nicht beteiligten – abgelehnt. Hier kann der Einsatz von objektiven arbeitspsychologischen Analyseverfahren durch professionelle Arbeitsgestalter Abhilfe schaffen.

Bei der Auswahl von Prototypingwerkzeugen ('interface-development-tools') kann die Checkliste von Hix und Schulman (1991) behilflich sein.

4.10.3 Stärken und Schwächen von Prototyping

Nachweisbare Stärken des Prototypingansatzes sind

+ Reduktion der Kostenüberschreitungen,
+ Reduktion der Terminüberschreitungen,
+ schnellere Verfügbarkeit der Software,
+ besseres Verständnis der Benutzer für die Möglichkeiten und Grenzen der Anwendung,
+ benutzungsgerechte Lösungen, insbesondere für die Dialog- und Anwendungsschnittstelle.

Bekannte Schwächen, welche sich jedoch bei adäquatem Vorgehen vermeiden lassen, sind

– Quick-and-dirty-Entwicklungen statt sauber programmierte Lösungen,
– zu langes Festhalten am Prototyp und dem einmal eingeschlagenen Weg,
– bei komplexen Applikationen ist kein effektives Prototyping möglich,

– Qualifizierungserfordernisse für die Analytiker bzw. Programmierer hinsichtlich der Prototypingwerkzeuge,

– unzureichende Prototypingwerkzeuge (siehe Übersicht 2 im Anhang).

4.11 CHECKLISTEN

Checklisten sind speziell für die Bewertung der Benutzungsfreundlichkeit entwickelt worden und zumeist nur in englischen Fassungen erhältlich. Deutschsprachige Checklisten sind in Spinas et al. (1983), Baitsch et al. (1989), Duell und Katz (1990), Koch, Reiterer und Gärtner (1990) sowie Siemens Nixdorf Styleguide (1992) zu finden (siehe auch Übersicht 2 im Anhang).

Stärken:

+ Gegenüber Benutzungstests verringert sich der Bewertungsaufwand beim Einsatz von Checklisten beträchtlich, womit das wirtschaftliche Interesse an diesen Verfahren gegeben ist.

+ Checklisten helfen zu verhindern, dass wichtige Aspekte bei der Bewertung vergessen werden.

+ Checklisten eignen sich besonders als Hilfsmittel bei der Bearbeitung von Routinefällen.

+ Checklisten enthalten in komprimierter Form die als relevant anzusehenden Aspekte.

Schwächen:

– Checklisten können Sachverstand nicht ersetzen; das Arbeiten mit ihnen bedingt hinreichende Kenntnis von Aufgaben, Problemen, Zusammenhängen und Lösungsansätzen.

– Die Zuverlässigkeit ist geringer als bei anderen Verfahren, da bewusstes oder unbewusstes Missverstehen der Fragen möglich ist.

– Nur begrenzte Möglichkeiten zur eingehenderen Erläuterung der einzelnen Fragen sind bei Papierversionen gegeben.

– Spezielle Aspekte, die nicht in das Raster der Checkliste hineinpassen, lassen sich nicht adäquat bei der Auswertung berücksichtigen.

4.12 EVALUATIONSTECHNIKEN

Bewertungs- und Evaluationstechniken sind im vorliegenden Zusammenhang formalisierte Verfahren, die die zu beurteilenden Systeme hinsichtlich ihrer Eigenschaften in eine Rangordnung bez. der angestrebten Benutzungseigenschaften bringen. Diese Rangordnung kann dann als Ausgangsbasis für eine Entscheidung dienen. Die Formalisierung der Bewertungsverfahren strebt folgende Ziele an:

(1) Die Vergleichbarkeit der Ergebnisse über alle bewerteten Systeme hinweg,

(2) die Nachvollziehbarkeit des Bewertungsvorganges.

Die verschiedenen Bewertungstechniken lassen sich hinsichtlich

(1) ihres Aufwandes,

(2) ihres Formalisierungsgrades und

(3) der Messbarkeit der verwendeten Bewertungsdimensionen unterscheiden.

Für eine Übersicht über die verschiedenen Reviewverfahren und ihre Handhabung sei auf Freedman und Weinberg (1990) sowie Frühauf, Ludewig und Sandmayr (1991b) verwiesen. Die Beteiligung von Endbenutzern bei Reviewsitzungen wird von Freedman und Weinberg (1990) explizit gefordert. Die Art der Beteiligung hängt vom Review und der fachlichen Qualifikation der Beteiligten ab.

Mit dem EVADIS-Verfahren (Oppermann et al. 1992) steht ein Instrument zur Verfügung, mit dem die Qualität einer Benutzungsoberfläche bewertet werden kann. Es handelt sich dabei um einen Fragenkatalog, der verschiedene Prüfkriterien enthält. Fragen werden nach einem festgelegten Ablauf beantwortet und ausgewertet. EVADIS-II wurde mit dem Ziel entwickelt, Softwaresysteme für den Büro- und Verwaltungsbereich zu evaluieren. Die Benutzer sind vornehmlich "Arbeitende in diesem Bereich, wie beispielsweise Schreibkräfte, Sachbearbeiter oder Wissensverarbeiter (Fach- oder Führungskräfte)". Es kann auf Softwareprodukte wie Datenbanken, Tabellenkalkulation, Bürografik, Textverarbeitung und elektronische Post angewendet werden.

Das EVADIS-II-Verfahren versteht sich als "Versuch, die relevanten ergonomischen Eigenschaften von Dialogsystemen für den Büro- und Verwaltungsbereich in einer ganzheitlichen Weise zu erfassen". Ganzheitlich me-

int, dass "möglichst alle benutzer- und aufgabenrelevanten Eigenschaften der Schnittstelle in allen Teilen des fraglichen Arbeitssystems" ermittelt werden sollen. "Es geht also nicht nur um die Erwartungskonformität einer Schnittstelle, nicht nur um ihre Steuerbarkeit, um zwei allgemeine Kriterien für Schnittstellengüte zu nennen. Es geht auch nicht allein um die Leistungsfähigkeit und Güte einzelner Systemkomponenten, wie z.B. eines Hilfesystems. Ganzheitlichkeit bedeutet auf der anderen Seite allerdings auch nicht, dass das Verfahren beanspruchen könnte, Schnittstellengüte abschliessend und erschöpfend darzustellen." (Oppermann et al. 1992)

Für EVADIS-II sind hauptsächlich die folgenden drei Anwendungsmöglichkeiten vorgesehen:

(1) *Bewerten während der Systemgestaltung:*
Einbeziehung von EVADIS in den Entwicklungsprozess eines entstehenden Softwareproduktes für ein Bürosystem. Beispielsweise beim Einsatz von Prototyping. Mögliche Anwender sind Softwarehäuser oder EDV-Abteilungen.

(2) *Bewerten von Systemausprägungen:*
Bewertung eines fertigen Softwareproduktes für ein Bürosystem hinsichtlich der ergonomischen Qualität einzelner Systemausprägungen, beispielsweise im Rahmen eines Abnahmetests. Mögliche Anwender sind Auftraggeber von Software oder Softwareprüfstellen.

(3) *Bewerten zum Zwecke eines Systemvergleichs oder einer Marktorientierung:*
Bewertung mehrerer fertiger Softwareprodukte für ein Bürosystem im Rahmen eines Systemvergleiches. Beispielsweise vor einer Kaufentscheidung von Standardsoftware. Mögliche Anwender sind Auftraggeber von Software, Softwareprüfstellen, Konsumentenberatungseinrichtungen oder Technologieberatungsstellen von Gewerkschaften.

Der Einsatz des EVADIS-Verfahrens setzt hinreichende Systemkenntnisse der zu evaluierenden Software voraus.

4.13 CODEINSPEKTION

Da in der Realisierungsphase das zu erstellende System ausprogrammiert wird, ist die Beteiligung der Endbenutzer auf ein Minimum reduziert. Als Beteiligungsmethoden kommen Reviewsitzungen mit Codeinspektion (Freedman und Weinberg 1990) in Frage. Dabei können die Benutzer – bei entsprechender fachlicher Qualifikation – neben der Beurteilung der Ver-

ständlichkeit und Korrektheit des erzeugten Source- und Objektcodes auch für die Zusammenstellung der benötigten Testdatensätze mitverantwortlich sein.

Obwohl – im Idealfall – ein ausreichender Aufwand in den frühen Phasen investiert worden ist, kann es dennoch vorkommen, dass offene Fragen und Probleme erst bei der Realisierung auftreten. Hierbei hat es sich als besonders nützlich erwiesen, wenn möglichst grosse Teile der Entwicklung am Ort der Arbeitsstätte durchgeführt werden können, um somit möglichst schnelle Rückfragen zwischen Benutzern und Entwicklern zuzulassen. Die Ergebnisse dieser Rückfragen sollten festgehalten und in den nächsten Reviewsitzungen angesprochen werden.

4.14 BENUTZUNGS- UND USABILITY-TESTS

Benutzungs- bzw. Usabilitytests sind eine Methode, um speziell die *interaktiven Eigenschaften* von Softwareprodukten zu erfassen. Sie lassen sich in zwei Klassen unterscheiden: *aufgabenorientierte* und *benutzungsorientierte* Tests (siehe auch Dumas und Redish 1993 sowie Nielsen 1993). Die rein technischen Tests werden nicht weiter besprochen (siehe hierzu Frühauf et al. 1991b). Die aufgabenorientierten Benutzungstests sind bei der Evaluation eines (z.B. vertikalen) Prototyps zur Gewinnung von Gestaltungs- und Verbesserungsvorschlägen bzw. zur Analyse von Schwachstellen in der Benutzbarkeit einsetzbar. Jeder Test wird durch einen oder zwei Testleiter vorbereitet und durchgeführt. Es empfiehlt sich, dass ein Produktverantwortlicher und ein Repräsentant der Entwicklungsabteilung als passive Beobachter beteiligt sind, um die Vermittelbarkeit der Ergebnisse zu gewährleisten.

Benutzungstests werden in der Regel in einem speziell eingerichteten Labor durchgeführt (siehe Nielsen 1994). Sie können aber auch direkt am Arbeitsplatz erfolgen. Benutzungsorientierte Tests lassen sich in zwei Gruppen unterteilen:

Induktive ('formative evaluation') und *deduktive* ('summative evaluation') Benutzungstests.

Die induktiven Benutzungstests sind bei der Evaluation einer (Vor-)Version zur Gewinnung von Gestaltungs- und Verbesserungsvorschlägen bzw. zur Analyse von Schwachstellen in der Benutzbarkeit einsetzbar. Induktive Benutzungstests können immer dann zum Einsatz kommen, wenn nur *eine* Version der zu testenden Software vorliegt. Demgegenüber verfolgen de-

duktive Benutzungstests primär den Zweck, zwischen mehreren Alternativen (mindestens zwei Prototypen bzw. Versionen) zu entscheiden. Zusätzlich lassen sich jedoch auch mit deduktiven Benutzungstests Gestaltungs- und Verbesserungsvorschläge gewinnen.

4.14.1 Aufgabenorientierter Benutzungstest

Wie bei den Szenarien wird bei aufgabenorientierten Benutzungstests eine möglichst realistische Arbeitssituation nachgestellt. Im Unterschied zu den Szenarien handelt es sich nicht um eine verbale Beschreibung, sondern um das möglichst realistische Nachbilden der jeweiligen Arbeitssituation (siehe auch die 'use cases' bei Booch 1994). Die aufgabenorientierten Benutzungstests lassen sich z.B. in Benutzer-Entwickler-Workshops einsetzen, um eine typische Arbeitssituation im Rollenspiel nachzuspielen. Dadurch kann dem Entwickler eine direkte, anschauliche Rückmeldung über die zunächst verbal beschriebene Situation gegeben werden. Ein Benutzer übernimmt dabei seine Rolle als Mitarbeiter in einer typischen bzw. geplanten Arbeitssituation mit entsprechend abgesprochenen Aufgaben.

Unter Einsatz des erstellten Prototyps wird die Aufgabenbearbeitung durchgespielt. Wenn noch kein Prototyp vorhanden ist, kann dieser auch mit möglichst einfachen Hilfsmitteln simuliert werden (Pappschachtel als Computerattrappe, Folien oder Papierblätter als Maskenersatz usw.; siehe Ehn und Kyng 1991). Die anderen Benutzer übernehmen neben dem Testleiter die Rolle der Beobachter und Kommentatoren, um z.B. Unterschiede zu der von ihnen bevorzugten Bearbeitungsweise festzustellen. Am besten lässt sich ein aufgabenorientierter Benutzungstest direkt am Arbeitsplatz oder in unmittelbarer Nähe dazu durchführen.

4.14.2 Induktiver Benutzungstest

Bei der Durchführung eines induktiven Benutzungstests ist eine Anzahl von Bedingungen zu beachten. Da die meisten Bedingungen zur Durchführung eines induktiven Benutzungstests mit denen eines deduktiven Benutzungstests übereinstimmen, werden bei der Darstellung deduktiver Benutzungstests nur noch die Änderungen und Ergänzungen erwähnt. Um Artefakte bei der Benutzung – hervorgerufen durch unvollständige Systemfunktionalität – zu vermeiden, sollte im Bereich der zu testenden Systemfunktionen ein möglichst realitätsgerechtes Systemverhalten gegeben sein. Um die einzelnen Aktionen der Benutzer später auswerten zu können, empfiehlt es

sich, eine automatische Aufzeichnung der Tastendrucke in das Testsystem einzubauen.

Kasten 20: Beispiele für problemorientierte und handlungsorientierte Beschreibungen von Testaufgaben.

> Problemorientierte Aufgabenbeschreibung:
> "Erstellen Sie bitte einen Brief mit folgendem Inhalt ... und bereiten Sie ihn zum Eintüten und Versenden an folgende Adressen ... vor. Benutzen Sie dabei bitte als Briefvorlage das Dokument mit dem Namen ..."
>
> Handlungsorientierte Aufgabenbeschreibung:
> "Laden Sie bitte das Textdokument mit folgendem Namen ... und ergänzen Sie es um den folgenden Inhalt ... Versehen Sie dieses Textdokument mit allen für die Serienbrieferstellung notwendigen Steueranweisungen ... usw."

Als erstes sind eine oder mehrere Testaufgaben abgestimmt auf die zu testenden Systemteile festzulegen. Diese Testaufgaben sind dem typischen Aufgabenkontext des zukünftigen Endbenutzers zu entnehmen. Sofern dieser Aufgabenkontext nicht oder nur vage bekannt ist, sollten die Testaufgaben zumindest jedoch typische Teilaufgaben enthalten. Eine einzelne Testaufgabe sollte nicht zu komplex, aber auch nicht zu einfach sein; die Bearbeitungszeiten sollten ungefähr zwischen fünf und dreissig Minuten liegen. Die Formulierung der Testaufgaben sollte den Benutzern während der Aufgabenbearbeitung als schriftliche Fassung zugänglich sein. Die Aufgabenbeschreibung hat das unterschiedliche fachliche Vorwissen der Benutzer zu berücksichtigen; bei hohem fachspezifischen Vorwissen ist die Beschreibung möglichst *problemorientiert* abzufassen, bei geringem fachspezifischen Vorwissen ist die Beschreibung *handlungsorientiert* zu halten (siehe Kasten 20). Die handlungsorientierte Aufgabenbeschreibung soll verhindern helfen, dass die beobachteten Benutzungsprobleme überwiegend aufgrund fehlenden Fachwissens zustande gekommen sind.

Im Unterschied zum aufgabenorientierten Benutzungstest sind beim induktiven Benutzungstest *mehrere* Benutzer als Softwaretester beteiligt. Die Benutzergruppe sollte möglichst *repräsentativ* für die Population der Endbenutzer sein. Die Auswahl der Benutzer kann z.B. *zufällig* aus dem poten-

tiellen oder aktuellen Benutzerkreis erfolgen (für weitere Stichprobentechniken siehe Bortz 1984, Kapitel 4). Die ausgewählten Benutzer sollten das zu testende System nicht kennen. Da sich diese Bedingung bei grösseren Entwicklungsphasen bzw. Weiterentwicklungen oftmals nicht einhalten lässt, muss die Vorerfahrung der Benutzer mit dem System bzw. ähnlichen Systemen kontrolliert werden. Die Gruppengrösse sollte aus statistischen Gründen zwischen sechs und zwanzig Personen liegen (zur Abschätzung der Effektgrösse und Testpersonenanzahl siehe Bortz 1984, Kapitel 6). Wenn kein oder nur sehr wenig Wissen über die Population der Endbenutzer vorliegt, lohnt es sich, die Benutzergruppe hinsichtlich der folgenden Parameter möglichst *heterogen* zusammenzusetzen: EDV-Vorerfahrung, Alter, Geschlecht, Ausbildung, Beruf, zu unterstützende Aufgaben.

Anzustreben ist eine den realen Einsatzbedingungen möglichst entsprechende Beobachtungsumgebung. Da es für die spätere Auswertung jedoch sehr wichtig ist, das Verhalten der einzelnen Benutzer sowie das entsprechende Systemverhalten auf Video, Tonband, 'screenrecording' usw. aufzuzeichnen, empfiehlt es sich, einen ruhigen, abgeschlossenen Raum mit entsprechendem Mobiliar zu benutzen. Stehen keine speziellen Aufzeichnungsgeräte zur Verfügung, sind für den Testleiter einfach auszufüllende Protokollbogen (z.B. mittels Strichlisten für vorgegebene Problem-, Fehlerkategorien usw.) sehr nützlich.

Der Testleiter sollte sich stets darum bemühen, jedem Benutzer das Gefühl der Wichtigkeit und Wertschätzung zu vermitteln. Die folgenden drei Aspekte sind für die Schaffung eines vertrauensvollen und für Kritik offenen Klimas hilfreich.

(1) Mit möglichst allgemein verständlichen Worten wird dem Benutzer Ziel und Zweck des Benutzungstests erläutert (z.B. Test eines Prototyps in einem frühen Entwicklungsstadium usw.).

(2) Es ist besonders wichtig, dem Benutzer verständlich zu machen, dass das System und *nicht* er als Benutzer getestet werden soll. Jede Art von Schwierigkeiten seitens des Benutzers bei der Aufgabenbearbeitung sind von besonderem Interesse!

(3) Jedem Benutzer muss zugesichert werden, dass er die Durchführung des Benutzungstests jederzeit unterbrechen und sogar abbrechen kann, ohne dass irgendwelche negativen Konsequenzen (z.B. Wegfall einer zugesagten Bezahlung usw.) erfolgen. Die vollständige Freiwilligkeit

der Teilnahme und Zusicherung der Vertraulichkeit ist unabdingbar für die Gewinnung von brauchbaren Beobachtungen.

Jedem Benutzer werden alle für die Durchführung des Benutzungstests in dem Beobachtungsraum vorhandenen Geräte (Computer, Videokamera, Mikrophone usw.) hinsichtlich Einsatzzweck erläutert. Als besonders wichtig erweist sich eine möglichst standardisierte Einführung in die Handhabungs- und Bedienungsweise der zu testenden Software. Je nach Vorerfahrung des Benutzers können unterschiedliche Schwerpunkte gesetzt werden. Um die Motivation der Benutzer bei einer längeren Einführungsphase (in die Benutzung der Software) aufrechtzuerhalten, hat es sich als sinnvoll erwiesen, den Benutzer zur Generierung von Frage- und Problemstellungen über die 'Welt der Anwendungsobjekte' zu stimulieren und zur selbständigen Exploration anzuregen. Dennoch muss unbedingt darauf geachtet werden, dass jeder Benutzer das gleiche Vorwissen erhält. Da diese Bedingung nur schwer zu kontrollieren ist, begnügt man sich oftmals mit Personen *ohne* jegliches EDV-Vorwissen.

Um die Handlungsziele der Benutzer während der Aufgabenbearbeitung erfassen zu können, werden sie gebeten, 'laut zu denken' (siehe zur Methode Ericsson und Simon 1984, Moll 1987). Viele Benutzer haben jedoch Schwierigkeiten, ihre Gedanken während der Bearbeitung einer Aufgabe laut zu äussern. Es empfiehlt sich daher, dem Benutzer zu verdeutlichen, was mit 'lautem Denken' gemeint ist, und mit ihm einige Übungsbeispiele gemeinsam durchzugehen. Trotzdem neigen Benutzer einerseits bei hochroutinisierten Handlungen, andererseits bei komplexen Problemlösungsprozessen dazu, das 'laute Denken' einzustellen. Dies kann man durch folgende Massnahmen umgehen:

(1) Der Testleiter fordert den Benutzer in solchen Situationen zum 'lauten Denken' auf.

(2) Es werden *zwei* Benutzer als Paar mit der Aufgabenbearbeitung betraut, wodurch die verbale Kommunikation zwischen beiden Partnern das 'laute Denken' ersetzt.

(3) Man führt nach Beendigung des Benutzungstests dem Benutzer problematische Ausschnitte per Videoaufzeichnungen vor und bespricht mit ihm seine jeweiligen Handlungsabsichten und die dabei aufgetretenen Probleme (die Videokonfrontationsmethode, siehe Mittenecker 1987).

Es empfiehlt sich, vor Beginn einer Testserie ein bis zwei Probedurchläufe zur Optimierung der gesamten Testprozedur durchzuführen. Als bedeutsa-

me Einflussgrössen auf die Art und Weise der Aufgabenbearbeitung haben sich die EDV-Vorerfahrung der Benutzer und die Hilfestellungen durch den Testleiter herausgestellt. Beide Variablen sollten erfasst werden, um sie später bei der statistischen Auswertung berücksichtigen zu können. Die Vorerfahrung lässt sich mittels Fragebogen erfassen. Die Art und die Anzahl der gegebenen Hilfestellungen des Testleiters wird von diesem während des Benutzungstest auf einem Protokollbogen vermerkt.

Als Testkriterien werden alle erhobenen Messwerte bezeichnet, welche bei der Auswertung Aufschluss über die Güte der Benutzbarkeit des zu testenden Systems Auskunft geben können. Die Menge der Testkriterien teilt sich auf in die Menge der *qualitativen* Messgrössen (problematische Benutzungssituationen, Fehlerarten usw.) und die Menge der *quantitativen* Messgrössen auf (Bearbeitungsdauer, Anlernzeit, Überlegungszeit, Anzahl Tastendrucke, Anzahl Fehler; siehe Dumas und Redish 1993).

Für die Gestaltung des zeitlichen Rahmens gibt es grundsätzlich zwei verschiedene Vorgehensweisen:

(1) Der Benutzer wird aufgefordert, die gestellte Aufgabe solange zu bearbeiten, bis er sie vollständig gelöst hat oder die Bearbeitung auf eigenen Wunsch abbricht.

(2) Dem Benutzer wird eine maximale Zeitdauer für die Aufgabenbearbeitung zur Verfügung gestellt.

Falls die Aufgabenbearbeitungszeit als ein relevanter Indikator für Benutzungseigenschaften des zu testenden Systems herangezogen werden soll, empfiehlt sich die erste Variante. Bei der zweiten Variante muss man für die Auswertung den erreichten Lösungsgrad der Aufgabe bewerten. Diese Bewertung ist oft nicht einfach möglich.

Der Testleiter hält sich während der Aufgabenbearbeitung ruhig im Hintergrund. Für ihn ist es sehr wichtig, seinen Wunsch, dem Benutzer in problematischen Situationen sofort zu helfen, zurückzuhalten. Hier kann eine klare Absprache zwischen Testleiter und Benutzer hilfreich sein: Nur wenn sich der Benutzer explizit an den Testleiter wendet und um Hilfe nachfragt, wird der Testleiter aktiv. Ein zu frühes Eingreifen des Testleiters hindert den Benutzer, eine eigene Lösung zu finden. Als Testleiter sollten daher keine an der Entwicklung des zu testenden Systems unmittelbar beteiligten Personen eingesetzt werden. Es wird sogar verschiedentlich empfohlen, um eine möglichst realistische Beobachtungssituation herzustellen, dass der Testleiter sich völlig passiv verhält und dies dem Benutzer vorher deutlich

macht. Weitere wertvolle Hinweise zum Testleiterverhalten findet man in Gniech (1976, S. 41-57).

Jedem Benutzer muss die Gelegenheit gegeben werden, sofern dies nicht schon im Rahmen der Videokonfrontation eingeplant ist, für ihn wichtige und noch offene Fragen zu Zwecken und Zielen des Benutzungstests, problematische Situationen während der Aufgabenbearbeitung usw. in einem Gespräch nach Beendigung der Aufgabenbearbeitung mit dem Testleiter klären zu können. Meistens erhält man aus diesen Gesprächen sehr zutreffende Problemsichten. In dieser Nachbereitung lassen sich auch Fragebogen mit standardisierten oder offenen Fragen zur Beantwortung spezieller Benutzungsaspekte einsetzen.

Während der Durchführung eines Benutzungstests wird man immer wieder überraschende und sehr informative Benutzungsweisen ('critical incidents', 'Stolpersteine') beobachten können. Es lohnt sich, diese auf Video (am besten Super-VHS wegen der höheren Bildauflösung) aufzuzeichnen, um sie dann mit den Entwicklern später detailliert diskutieren zu können. Besser ist es jedoch oftmals, die Entwickler – oder einen Repräsentanten durch eine Einwegscheibe vom Testraum getrennt dem Test beiwohnen zu lassen. Wichtig ist dabei, dass die beobachtbaren Schwierigkeiten niemals dem Benutzer, sondern ausschliesslich der Benutzbarkeit des zu testenden Systems angelastet werden. Wenn mehrere Benutzer dieselben Schwierigkeiten hatten, kann es nicht an ihnen liegen!

Die aufgezeichneten Beobachtungsdaten (Super-VHS-Video, Tonband, Logfile, 'screenrecording') müssen hinsichtlich der designrelevanten Informationen ausgewertet werden. Dazu empfiehlt es sich, auf der Grundlage der durchgeführten Beobachtungen ein Auswertungsraster aufzustellen und dieses Raster systematisch auf die Beobachtungsdaten aller Benutzer anzuwenden. Aufwendig und schwierig sind die qualitativen Auswertungen von Logfiledaten, weil es sich hierbei oftmals um komplexe Mustererkennungsprozesse handelt, welche bisher nur begrenzt automatisch ausführbar sind. Mit dem Auswertungsprogramm AMME lassen sich einige relevante Messgrössen gewinnen (siehe hierzu Rauterberg 1993). Ein umfangreiches Messinventar zur Messung der Benutzungsfreundlichkeit wird vom National Physical Laboratory in England angeboten (MUSiC 1993).

Als eine besonders informative Quelle für designrelevante Entscheidungen hat sich die Analyse der Videoaufzeichnungen ergeben. Hierbei werden Erklärungsmodelle für bestimmtes Benutzungsverhalten aufgestellt, um das beobachtbare Verhalten in problematischen Situationen handlungspsycho-

logisch begründen zu können. Die eingehendere Beschäftigung mit diesen Situationen führt dann unter Berücksichtigung der oben beschriebenen Kriterien zu konstruktiven, tragfähigen Gestaltungsvorschlägen.

Um die in der qualitativen Auswertung gewonnenen Erklärungen für einzelne Benutzungsprobleme auf eine möglichst breite Basis zu stellen (Problem der 'Generalisierbarkeit'), sind statistische Auswertungen unumgänglich. Diese können von einfachen, deskriptiven Häufigkeitsauszählungen bestimmter Problemklassen bis hin zu komplexen inferenzstatistischen Zusammenhangsanalysen gehen. Dazu ist es notwendig, dass die verschiedenen Benutzungseigenschaften (Dauer der Aufgabenbearbeitung, mittlere Überlegungszeit, Fehlerart, Ausmass an Beanspruchung usw.) pro Benutzer in quantifizierter Form vorliegen.

4.14.3 Deduktiver Benutzungstest

Deduktive Benutzungstests dienen primär der Entscheidungsfindung zwischen verschiedenen Systemalternativen bzw. zur Kontrolle der erreichten Verbesserung und erst sekundär der Gewinnung von Gestaltungsvorschlägen. Es ergeben sich daher einige Unterschiede in den Anforderungen an die Durchführung von deduktiven Benutzungstests.

Im Gegensatz zu induktiven Benutzungstests müssen die verschiedenen zu testenden, alternativen Systemversionen beim deduktiven Benutzungstest verstärkt der Forderung nach einem – an realen Einsatzbedingungen gemessenen – realistischen Systemverhalten (das heisst, möglichst funktionale Vollständigkeit, adäquates Systemantwortzeitverhalten usw.) genügen. Diese Anforderungen sind deshalb wichtig, weil die Entscheidung zwischen den Systemalternativen primär mittels quantitativer Messgrössen gefällt wird.

Je nachdem, ob viele Benutzer wenig Zeit haben (Fall I) oder ob wenige Benutzer eher mehr Zeit haben (Fall II), um an einem Benutzungstest teilzunehmen, ergibt sich ein unterschiedliches Testdesign. Im Fall I wird pro Systemalternative eine eigene Gruppe gebildet. Im Fall II hat lediglich eine Gruppe alle Systemalternativen zu testen. Um im Fall I jedoch Lerneffekte zu kontrollieren, muss die Reihenfolge der zu testenden Systeme innerhalb dieser einen Gruppe ausbalanciert werden. Im Fall II sollten dann auch für jede Systemalternative *parallele* Testaufgaben erstellt werden.

Die Gruppengrösse sollte aus statistischen Gründen mindestens sechs Personen pro Gruppe betragen. Wenn man plausible Annahmen über den zu

erwartenden Oberflächeneffekt hinsichtlich einer quantitativ zu messenden Benutzungseigenschaft (z.B. Dauer der Aufgabenbearbeitung, Anzahl Fehler usw.) für die Alternativen hat, kann man die genau benötigte Gruppengrösse auch ausrechnen. Die einzelnen Gruppen müssen hinsichtlich verschiedener Parameter möglichst homogen sein: EDV-Vorerfahrung, Geschlecht, Alter, Beruf. Mittels inferenzstatistischer Auswertungsverfahren können die gewonnenen Beobachtungsergebnisse auf ihre *Generalisierbarkeit* hin getestet werden.

4.15 SCHULUNG

Im folgenden Abschnitt geht es um die Schulung der Benutzer im Umgang mit dem neuen EDV-System.

Warum

Die Schulung der Benutzer wird in der Praxis leider oft vernachlässigt; es genügt keinesfalls, die Mitarbeiter mit einem oder mehreren Handbüchern ihrem Schicksal zu überlassen! Die Folgen sind in der Regel überforderte und frustrierte Benutzer; ursprünglich vorhandene positive Einstellungen zu Computersystemen können sich in Ablehnung und Abneigung umwandeln, Ängste vor Bedienungsfehlern können entstehen, letztlich kann die Systembenutzung überhaupt verweigert werden. Um diese Folgen zu vermeiden und den Benutzern eine effiziente Beherrschung der Systemfunktionalität zu ermöglichen, ist eine seriöse Schulung unumgänglich!

Wer

Grundsätzlich sollen *alle* Benutzer durch eine gründliche Ausbildung auf das neue System vorbereitet werden. Das heisst, dass auch gelegentliche Benutzer – wie z.B. Vorgesetzte, die das System nur einmal wöchentlich zur Abfrage von Statistiken benutzen – zumindest minimale Kenntnisse über den Umgang mit dem System erhalten sollten; sie werden dadurch unabhängiger von Hilfeleistungen geübter Benutzer.

Was

Je besser die Ausbildung, desto besser ist auch die Nutzung der Systempotentiale! Die Vermittlung von *Bedienungskenntnissen* (sogenannte *operative Kenntnisse*) *allein* ist *ungenügend!* Es hat sich als sehr wichtig gezeigt, dass den Benutzern zusätzlich auch *funktionale Kenntnisse* über den Computer und die Software vermittelt werden. Erst diese Kenntnisse ermögli-

chen den Benutzern, sich ein realistisches gedankliches Modell – das heisst, ein vorstellungsmässiges Abbild – der Funktionen und Zusammenhänge des Systems aufzubauen.

Beispielsweise wird der Umgang mit dem Computer erleichtert, wenn bekannt ist, was Arbeitsspeicher, Betriebssystem, Directory, Festplatte usw. sind. Dadurch verhilft man den zukünftigen Benutzern zur umfassenden Beherrschung des neuen Arbeitsmittels, was ein Verstehen der (z.B. durch Tastendruck ausgelösten) Verarbeitungsprozesse einschliesst. Das Gefühl der Unterlegenheit gegenüber einer nicht durchschaubaren Maschine sowie die Angst vor fehlerhaften Eingaben und deren Auswirkungen können damit weitgehend abgebaut werden. Gute gedankliche Modelle erleichtern darüber hinaus auch die Umstellung bei späteren Erweiterungen oder Systemwechseln, da die neu zu lernenden operativen Kenntnisse sich bei ähnlichen Funktionen besser aneignen lassen.

Inhalt der Schulung:

(1) Grundlagen der EDV,

(2) das EDV-System im Betrieb: Ebenen, Bereiche, Komponenten und deren Funktionen,

(3) funktionale Kenntnisse: Funktionen der Software und deren Einsatz im Arbeitsablauf, Unterschiede zur früheren Arbeitsweise,

(4) operative Kenntnisse: Bedienung der Geräte und Programme, Verhalten bei Fehlern, Systemabstürzen,

(5) Benutzung von Handbüchern, On-line-Tutorials.

Wie

Die erste Schulung findet vorzugsweise in Form eines ein- bis mehrtägigen, betriebsinternen Kurses mit Übungsmöglichkeiten am Bildschirm statt. Eine Mischung von Theorie mit eingestreuten praktischen Übungen des Gelernten hat sich bewährt. On-line-Tutorials können der individuellen Vertiefung – im Selbststudium – und Einübung dienen, sollten aber keinen Ersatz für 'traditionelle' Schulung mit einem Lehrer bilden, da bei Unklarheiten und Rückfragen ein Lehrer bessere Erklärungsmöglichkeiten hat (siehe auch Abschnitt 'Übungsarbeitsplatz'). Das Lernen in Kleingruppen mit gegenseitigen Unterstützungsmöglichkeiten ist empfehlenswert. Zwei Aspekte sind bei der Schulung besonders zu berücksichtigen:

(1) Aufgabenbezug
Das Lernen sollte im Kontext der konkreten, eigenen Aufgaben und Arbeitsabläufe erfolgen; dadurch werden die Einsatzmöglichkeiten der Software und deren Nutzen bei der Aufgabenerfüllung besser erkennbar. Ausserdem wird das Lernen von überflüssigem Ballast vermieden. Dies kann zur Folge haben, dass für Mitarbeiter verschiedener Abteilungen oder mit unterschiedlichen Aufgaben (z.B. Sachbearbeiter, Sekretär, Vorgesetzter) auch verschiedene Kurse mit dementsprechenden Anwendungsschwerpunkten durchgeführt werden. Aufgabenorientierte Schulung beinhaltet ein Ausrichten der Ausbildungsinhalte und Übungen auf die spezifischen Aufgaben einer Mitarbeitergruppe oder Organisationseinheit. Dies setzt voraus, dass auch die Ausbilder nicht nur über Systemkenntnisse, sondern auch über die jeweiligen Fachkenntnisse verfügen.
Der Erfolg derartiger Kurse rechtfertigt ihren vergleichsweise hohen Aufwand! In externen Kursen ist dieser Bezug auf die konkreten Aufgaben kaum möglich; ihr Wert für die Vermittlung von allgemeinem Grundlagenwissen über den Umgang mit einem Programm für Anfänger oder Spezialkenntnissen zur Ausreizung des Systempotentials für versierte Benutzer sei hier keineswegs angezweifelt. Betriebliche Unterstützung bei der Umsetzung des Gelernten auf die eigene tägliche Arbeit erweist sich aber in der Regel für viele Benutzer als hilfreich! Ferner besteht bei einigen Anbietern von externen Kursen die Möglichkeit der Abstimmung von Standardkursen auf betriebliche Bedürfnisse, sofern die ganze Klasse aus Mitgliedern der Belegschaft besteht.

(2) Den Kenntnissen angepasste Schulung
Um Überforderung der einen Teilnehmer und Unterforderung anderer Teilnehmer zu vermeiden, sollte die Schulung dem Ausbildungsstand und den Erfahrungen der Teilnehmer angepasst sein. Analog zu einer Skischule sollten die Auszubildenden in Gruppen von Teilnehmern mit gleichen bzw. ähnlichen Kenntnissen und Vorerfahrungen eingeteilt werden. Damit kann der Lernfortschritt für die meisten Teilnehmer optimal gefördert werden.

Übungsarbeitsplatz:

Zum Zwecke der spielerischen und damit angstfreien Gewöhnung an ein neues technisches System hat sich in der Praxis folgendes Vorgehen bewährt: Einige Zeit vor der Einführung wird ein Übungsarbeitsplatz eingerichtet, der ein Abbild der geplanten Arbeitsplätze darstellt, dessen techni-

sches System zur Vermeidung nachteiliger Auswirkungen infolge Fehlbedienung aber mit simuliertem Datenmaterial arbeitet. Den Mitarbeitern wird – nach einer angemessenen Schulung – freier Zugang zu diesem neuen Arbeitsplatz gewährt, so dass sie selbständig die Handhabung und Möglichkeiten des neuen Arbeitsmittels spielerisch erkunden können, da sie ja bei Fehlern keine gravierenden Folgen zu befürchten haben. Ein weiterer Vorteil dieses Vorgehens besteht in der frühzeitigen Entdeckung eventuell vorhandener Nachteile des neuen Arbeitsmittels, wodurch Verbesserungsvorschläge in der Realisationsphase berücksichtigt werden können.

Kasten 21: Beispiel Schneeballprinzip und mögliche Gefahren.

Drei Monate vor der Einführung eines Informationsabfragesystems in einer Versicherung wurde die Ausbildung der Mitarbeiter, die nach dem *Schneeballprinzip* erfolgen sollte, in Angriff genommen. Da gerade Ferienzeit war (Personalengpass, viele Diebstähle, Unfälle), mussten die Agenturen die am ehesten abkömmlichen Mitarbeiter delegieren. Diese wurden in einem halbtägigen Kurs rezeptartig mit der Bedienung des Bildschirmgerätes und den wichtigsten Abfrageprozessen vertraut gemacht. Da es noch keine Terminals gab, unterrichteten sie anschliessend ihre Arbeitskollegen anhand von Folienmaterial. Als das System dann etwa zweieinhalb Monate später eingeführt wurde, konnte kaum ein Mitarbeiter damit umgehen. Erst durch intensives Selbststudium des Benutzerhandbuches konnte die Unterstützung des Systems allmählich genutzt werden; einige Mitarbeiter waren aber schon so frustriert von ihren erfolglosen Bemühungen, dass sie die Arbeit mit dem Bildschirm vermieden und nach wie vor mit vorhandenen Archivinformationen arbeiteten.

'Schneeballprinzip':

Vielfach erfolgt die Schulung in Betrieben nach dem sogenannten '*Schneeballprinzip*'; nur ein kleiner Teil der Mitarbeiter besucht die Schulung und vermittelt nachher das erlangte Wissen und Können in der Arbeitsgruppe oder Abteilung selber weiter an die übrigen Mitarbeiter (siehe Kasten 21).

Dieses Beispiel zeigt u.a., dass das Schneeballprinzip eine sorgfältige Auswahl der zuerst Auszubildenden aufgrund ihrer bisherigen Erfahrung und weiteren Arbeit mit Computern, ihrer Motivation und didaktischen Fähig-

keiten sowie auch genügend Zeit erfordert. Eine vorangehende didaktische Schulung erweist sich vielfach als notwendig. Häufig werden diese Ausbilder dann auch aufgrund ihrer ausgeprägten Systemkenntnisse neben ihrer eigentlichen Aufgabe auch zu Koordinatoren bzw. Kontaktpersonen zwischen EDV- und Fachabteilung ernannt und bilden so eine dezentrale Anlaufstelle – lokale Experten – für Benutzer, die Fragen oder Schwierigkeiten im Umgang mit dem System haben. Ihre Rolle ist so wichtig, dass sie gegebenenfalls durch eine Entlastung von Arbeitsaufgaben unterstützt werden sollte, um ohne Überbelastung seriös wahrgenommen werden zu können.

Neben traditionellen Formen der Ausbildung hat sich gezeigt, dass das sogenannte 'exploratorische Lernen', bei welchem Benutzern die Möglichkeit zur eigenständigen Erkundung eines Systems mit einem Minimum an vorgegebenen Informationen geboten wird, zu guten Ergebnissen führt (siehe Dutke 1994).

Wann

Die Schulung muss mit der Systemeinführung zeitlich so koordiniert werden, dass das Gelernte laufend angewendet bzw. geübt werden kann. Erfolgt die Schulung – wie im Beispiel 'Schneeballprinzip' – zu früh, so gehen die operativen Kenntnisse infolge Nichtanwendung verloren; allenfalls können theoretische Grundbegriffe und funktionale Kenntnisse einige Zeit vor der Einführung vermittelt werden. Neben der Schulung in Kursen sollte für die Benutzer auch Literatur über das System wie z.B. verständliche, didaktisch aufbereitete Handbücher u.ä. verfügbar sein, damit sie im Selbststudium ihr Wissen erweitern bzw. vertiefen oder Gelerntes – aber bis anhin nicht Gebrauchtes – wiederholen können. Während der Einführungsphase sollte auch jederzeit ein Experte zumindest telefonisch erreichbar sein; in der Regel bevorzugen Benutzer gerade in dieser Phase menschliche Unterstützung gegenüber einer Unterstützung durch Handbücher, Hilfesysteme und Lernprogramme, weil ein Experte besser kontextbezogene, auf individuelle Bedürfnisse abgestimmte Hilfe leisten kann.

Eine seriöse Schulung bildet nicht nur die Basis für eine effiziente Systembenutzung; sie beeinflusst auch die Einstellung und das Verhalten der Benutzer bei zukünftigen Systemerweiterungen oder -veränderungen.

4.16 NACHEVALUATION BEIM SYSTEMEINSATZ

Mit der Einführung des Systems bzw. der Überführung in den produktiven Betrieb sind die Aufgaben der Systementwicklung noch nicht abgeschlossen, wie fälschlicherweise oft geglaubt wird. Erst die Erfahrungen im Alltagsbetrieb lassen eine endgültige Beurteilung zu, ob die angestrebten Ziele erreicht wurden bzw. welche Mängel – z.B. Details der Benutzungsoberfläche – noch vorhanden sind und korrigiert werden müssen. Ausführliche, gewissenhaft durchgeführte Programmtests vor der Systemeinführung sind zwar unabdingbar, können aber für einen Teil aller Geschäftsfälle höchstens logische Fehler oder Plausibilitätsfehler der Software ermitteln. Jedoch für die überall auftretenden und schwer vorhersehbaren Sonderfälle im Alltagsbetrieb zeigen sich eventuell noch vorhandene Schwachstellen und Defizite der Software! Deshalb ist eine Nachevaluation zirka drei Monate (die Dauer ist abhängig vom konkreten Projekt) nach der Systemeinführung angezeigt. Sie soll darüber Aufschluss geben,

(1) ob allgemein die gesetzten Ziele erreicht wurden bzw. weshalb nicht,

(2) ob die Software noch Fehler, z.B. in der Programmlogik, aufweist,

(3) inwieweit die angebotene Unterstützung von den Benutzern auch beansprucht wird bzw. ob wesentliche Systemfunktionen von den Benutzern – sei es mangels Kenntnis, fehlender Aufgabenangemessenheit oder infolge von Handhabungsproblemen – nicht genutzt werden, welche Mängel der Benutzungsoberfläche Benutzungsprobleme verursachen und

(4) ob für die vollumfängliche Nutzung der Systempotentiale ein zusätzlicher Schulungsbedarf besteht.

Methodisch gesehen bestehen folgende Möglichkeiten für eine derartige Nachevaluation:

(1) Messung des zeitlichen Aufwandes für einzelne Bearbeitungsgänge oder ganze Abläufe.

(2) Messung der Anzahl bearbeiteter Fälle im Vergleich mit der Situation vorher.

(3) Messung der Anzahl aufgetretener Fehler; qualitative Analyse der Fehler: Welche Fehler? Immer die gleichen Fehler?

(4) Protokollierung von Dialogabläufen mittels Logfiles, aus Gründen des Datenschutzes aber ohne die Erhebung persönlicher Benutzerdaten!

Derartige Protokolle können hilfreiche Hinweise für Schwachstellen – Stolpersteine – im Dialog geben; der Auswertungsaufwand kann allerdings beträchtlich sein!

(5) Persönliche Interviews mit und/oder schriftliche Befragung von Benutzern.

(6) Protokolle der Hotline oder einer sonstigen Supportstelle: Welche Fragen werden häufig gestellt, wo treten die meisten Probleme auf?

Anhand der Untersuchungsergebnisse wird dann zu entscheiden sein, ob und gegebenenfalls welche Änderungen, Korrekturen in der Software nötig sind bzw. ob ganze Module neu zu entwickeln sind. Sinnvollerweise sollten diese Änderungen vom Entwicklungsteam selbst und, weil der Einarbeitungsaufwand gross ist, *nicht* von einer generell mit Programmwartungsaufgaben beauftragten Gruppe durchgeführt werden. Die realisierten Änderungen sollten dann auch den Benutzern – on-line oder schriftlich – mitgeteilt werden. Erst nach einer derartigen Nachevaluation werden die Systemeffekte abschliessend beurteilbar sein. Der Einsatz wird so lange reibungslos funktionieren, bis aufgrund eines erhöhten Anspruchsniveaus der Benutzer oder eines gestiegenen Bedürfnisses nach zusätzlicher Unterstützung eine Weiterentwicklung erwartet wird.

5 Fazit

In diesem Teil B wurden eine Reihe von Methoden, Techniken und Verfahren vorgestellt, welche jeweils auf die eine oder andere Art eine Beteiligung von Benutzern ermöglichen. Für weiterführende Darstellungen zu diesen Methoden sei die angegebene Literatur empfohlen. Hauptanliegen dieses Abschnittes ist die Vermittlung grundlegenden Wissens zur Auswahl einzelner Methoden. Dabei haben wir das Schwergewicht auf die frühen Phasen (Quadrant-I und -II) gelegt. Fundiertes Wissen zur Anwendung dieser Methoden und Moderationstechniken wird in der angegebenen Literatur und entsprechenden Trainings- bzw. Schulungskursen vermittelt.

TEIL C: FALLBEISPIELE

Vorbemerkung zu den Fällen

In den folgenden fünf Fallbeispielen werden wir aufzeigen, wie sich die einzelnen Beteiligungsmethoden je nach Projekttyp praktisch einsetzen lassen, welche Ergebnisse dabei erzielt wurden und welche Probleme sich dabei im einzelnen ergeben haben.

Die geschilderten Fälle sind nicht erfunden, sondern stammen aus der Praxis! Aus didaktischen Gründen wurden sie vereinfacht. Besonders interessante Aspekte wurden hervorgehoben. Die Fälle sollen die in den bisherigen Kapiteln dargestellten Konzepte veranschaulichen. Sie zeigen auch die in der Praxis möglichen, vielfältigen Kombinationen von Formen, Vorgehensweisen und Methoden partizipativer Softwareentwicklung. Die Fälle können auch in Kursen für Übungszwecke verwendet werden.

Rahmenbedingungen	Fall 1: Versicherung	Fall 2: KBT	Fall 3: Bibliothek	Fall 4: ADI	Fall 5: PPS
Projekttyp	A	A	A	D	A
Projektkomplexität	hoch	hoch	mittel	mittel	sehr hoch
Projektinnovativität	gering	hoch	mittel	gering	mittel
Benutzergruppen	heterogen	heterogen	heterogen	sehr heterogen	heterogen
Aufgaben	anspruchsvoll	heterogen	Routineaufgaben	unbekannt	anspruchsvoll

Zitate ("...") sind wörtliche Aussagen beteiligter Personen.

Fall 1: Versicherung

1 AUSGANGSLAGE

In einer grossen Versicherung wurde der Beschluss zu einer Totalrevision des bestehenden, zirka 12 Jahre alten EDV-Systems gefasst. Die Revision sollte vor allem folgende Aspekte umfassen:

(1) Es sollten beträchtlich mehr aktuelle, fehlerfreie Informationen für tägliche Bearbeitungsvorgänge und für statistische Auswertungen verfügbar sein.

(2) Das System musste eine neue, benutzungsfreundliche Oberfläche aufweisen; dabei war insbesondere der Erweiterung von Suchmöglichkeiten der Benutzer nach Informationen anhand verschiedener Kriterien gebührend Rechnung zu tragen.

Das System sollte einem 'breiten' Benutzerkreis (Vorgesetzte unterschiedlicher Stufen; Sachbearbeiter; Sekretärinnen; angelernte Bürohilfskräfte) mit stark unterschiedlicher Computererfahrung und Benutzungshäufigkeit des Systems ermöglichen, mit geringem Aufwand umfassende Informationen über einen Geschäftsvorfall oder gezielt einzelne (Detail-)Informationen abzurufen; ausserdem sollte es zur Erfassung neuer Daten dienen. Das gleiche System sollte auch in verschiedenen Geschäftsbereichen (Departemente wie Policen, Schaden, Marketing usw.) mit dementsprechend unterschiedlichen Aufgaben sowie in Niederlassungen mit unterschiedlicher Organisation als technisches Hilfsmittel dienen. Die Heterogenität des Benutzerkreises, der Aufgaben und der Organisation erforderte also ein äusserst flexibel adaptierbares System, das in Zukunft rund 2500 Benutzern für zirka 80'000 Abfragen und 1000 Erfassungen pro Tag zur Verfügung stehen sollte. Obwohl das Vorhaben nicht sehr innovativ oder besonders komplex war, musste das System von Grund auf neu entwickelt werden, da wegen der genannten Anforderungen an eine Revision des bestehenden Systems nicht mehr zu denken war.

2 PROJEKTORGANISATION

Mit der Entwicklung des Systems wurde eine Projektgruppe beauftragt, die sich aus einem Analytiker (Projektleiter) und zwei Programmierern zusammensetzte; ergänzt wurde diese Gruppe durch einen internen Berater (Be-

nutzer-EDV-Koordinator mit langjähriger Erfahrung) sowie durch einen externen Berater für Fragen der Softwareergonomie und der Benutzerbeteiligung. Es wurde auch eine Benutzergruppe (N=11) gegründet, die sich stellvertretend für alle Benutzer an der Systementwicklung beteiligen sollte; die Aufgaben dieser Gruppe bestanden vorwiegend in der Formulierung von Anforderungen, der Beurteilung von Konzepten der Entwickler (im Sinne einer Vernehmlassung), der Weitervermittlung der Konzepte an die Arbeitskollegen und damit verbunden im Einholen und Weiterleiten von Feedback an die Entwickler.

Die Zusammensetzung der Benutzergruppe wurde im Hinblick auf die Repräsentativität der vertretenen Anwendungsbereiche bestimmt; zur Hauptsache waren Sachbearbeiter mit Sonderfunktionen und erweiterten EDV-Qualifikationen in der Gruppe vertreten (tägliche Benutzer mit viel Erfahrung). Die Benutzergruppe hatte in allen – ausser technischen – Fragen der Anwendungsentwicklung Mitbestimmungsrecht und entsprechende Mitverantwortung. Ihre Entscheidungen konnten nur noch vom Steuerungs- und Kontrollausschuss, dem die Projektgruppe Rechenschaft über die Projektarbeit leisten musste, geändert werden. Das geschah aber in der Regel kaum.

3 PROJEKTABLAUF

In einer Vorstudie wurden Benutzeranforderungen erhoben, Arbeitsabläufe untersucht, Schwachstellen eruiert und generelle Ziele abgeleitet. Zur Erhebung der Benutzeranforderungen wurden die Abteilungsleiter aller Departemente, die Leiter der grösseren Niederlassungen sowie einige Leiter der zur Hauptsache betroffenen Bereiche mündlich oder schriftlich nach ihren Wünschen für das zukünftige System befragt; ausserdem erhielten die Benutzervertreter die Gelegenheit, in einem zweitägigen Workshop ihre Erfahrungen mit dem alten System zu diskutieren und ihre Anforderungen an das neue System zu formulieren. Als Ergebnis dieser Erhebungen ergab sich eine Liste mit 280 sehr detaillierten und z.T. auch widersprüchlichen Benutzerwünschen, deren Auswertung der Projektgruppe einige Mühe bereitete. Parallel dazu wurden in 13 Abteilungen am Hauptsitz sowie in zwei Niederlassungen (damit wurde die Repräsentativität der verschiedenen Bereiche sichergestellt) verschiedenartige Arbeitsabläufe (z.B. Adressenerfassung; Informationssuche bei der Schadenbearbeitung, im Marketing usw.) grob untersucht. Die Analyse dieser Abläufe zeigte einige Schwachstellen

und führte – zusammen mit den Benutzeranforderungen – zur Formulierung genereller Ziele seitens der Projektgruppe:

(1) Möglichkeit individueller Ausnutzung umfangreicher, aktueller, fehlerfreier Daten,

(2) Benutzerfreundlichkeit der Systemhandhabung,

(3) Anpassbarkeit an unterschiedliche Ablauforganisation, bereinigte, in grundlegenden Aspekten aber vereinheitlichte Abläufe,

(4) Kundenorientierung.

Auf dieser Grundlage wurden von der Projektgruppe sodann in zahlreichen Workshops generelle Lösungskonzepte entwickelt und in verschiedenen, 'inoffiziellen' Vernehmlassungen (Kollegen) präzisiert; unterschiedliche Varianten wurden Nutzwertanalysen unterzogen (wobei quantifizierte, qualitative Aspekte eine wesentliche Rolle spielten), mit Statistiken (wo verfügbar) belegt und mit einer Empfehlung für eine Variante in einem Bericht dem Kontrollausschuss vorgelegt. In der Gesamtwirtschaftlichkeitsrechnung wurden lediglich die Kosten beziffert; der Nutzen bzw. Ertrag wurde explizit nicht mit Produktivitätssteigerungen ("das Projekt dient nicht Rationalisierungszwecken"), sondern mit qualitativen Verbesserungen angegeben.

Das Problem der Anpassbarkeit des Systems an unterschiedliche organisatorische Abläufe wurde mit dem Konzept der flexiblen, ablauforganisatorischen Dezentralisierung, das sich durch Veränderung weniger Systemparameter direkt in den Niederlassungen verwirklichen lässt, gelöst.

Der Benutzerfreundlichkeit des Systems wurde grosse Bedeutung beigemessen. So sollten die Transparenz, Kompatibilität und Aufgabenangemessenheit des Systems durch den Benutzereinbezug bei der Informationsorganisation (Verteilung der Informationen auf Bilder) und der Gestaltung von Dialogabläufen sichergestellt werden. Dem heterogenen Benutzerkreis wird durch das Angebot von zwei völlig unterschiedlichen Interaktionsmöglichkeiten Rechnung getragen: Im sogenannten 'Standardmodus' wird der ungeübte, sporadische Benutzer über Menüs und Eingabemasken zu den gewünschten Informationen geführt; auch in diesem Modus sind aber selbst für Erfahrene noch genügend Freiheitsgrade vorhanden, so dass Menüs übersprungen und Bilder direkt aufgerufen werden können.

Im sogenannten 'Expertenmodus' erfolgt die Interaktion mit dem System über Kommandos mit Parametern; dabei können – im Sinne einer Indivi-

dualisierung – häufig benutzte Abläufe als Makro abgespeichert werden. Die Umschaltung zwischen den beiden Modi ist einfach (ein Tastendruck), theoretisch könnten Benutzer also mehrmals täglich wechseln. Dies kann z.B. bei selten benutzten Funktionen eine Erleichterung auch für 'Experten' darstellen. Der Flexibilität des Systems (im Sinne von Sprungmöglichkeiten innerhalb und auch zu anderen Anwendungen) sowie der Minimierung von Dialogschrittfolgen durch Direktzugriff und 'kurze Dialogwege' kam nämlich in den Benutzeranforderungen grosses Gewicht zu. Schnittstellen zu anderen Anwendungen sollten vereinheitlicht sowie grundlegende Elemente (wie Belegung von Funktionstasten, Anordnung von Menüs, Meldungen usw.) in Übereinstimmung mit anderen Anwendungen gestaltet werden, soweit dies sinnvoll erschien (Problem der Konsistenz zwischen alten und neuen Anwendungen).

4 WEITERER PROZESSVERLAUF

Die Projektgruppe befasste sich im weiteren – neben konzeptueller Arbeit an EDV-technischen Fragen – mit der Gestaltung einzelner Masken und stellte schliesslich fest, dass dieses Vorhaben ohne die vorherige Beantwortung übergeordneter Fragen (Funktionalität, Informationsorganisation usw.) zum Scheitern verurteilt war. Daraufhin wurde die begonnene Arbeit unterbrochen und die Bearbeitung der erwähnten, übergeordneten Aspekte in Angriff genommen.

An einer Tagung wurden den Benutzervertretern Vorschläge zum Dialogkonzept und zur Informationsorganisation präsentiert sowie erste Reaktionen eingeholt. Als 'Hausaufgabe' wurde den Teilnehmern aufgetragen, sich an ihrem Arbeitsort im Kreise der Kollegen gründlich und kritisch mit den vorgestellten Konzepten auseinanderzusetzen sowie Kritik und Verbesserungsvorschläge schriftlich und mündlich der Projektgruppe zurückzumelden, was auch geschah. Befürchtungen der Projektgruppe, dass die Benutzervertreter einzelne Teilkonzepte bzw. deren Darstellungsform nicht verstehen würden, erwiesen sich als unbegründet. Allerdings zeigte sich, dass die Integration der Teile zu einem Ganzen lediglich anhand schriftlicher Vorlagen eine Überforderung der Teilnehmer darstellte. Insgesamt verlief diese erste Zusammenkunft in entspannter Atmosphäre, die Teilnehmer zeigten sich motiviert, mehrheitlich gut qualifiziert und ihrer Verantwortung bewusst. Ihre Zustimmung zu Konzepten war der Startschuss für die unabänderliche Programmierung eines Systems, das in den nächsten ca. zehn Jahren für 2500 Benutzer eine Arbeitsgrundlage darstellt.

Für die weiteren Arbeitsschritte war eine aktivere Mitarbeit der Benutzergruppe im Entwicklungsprozess vorgesehen. Ihr Beitrag sollte sich nicht mehr 'nur' auf Beurteilungsleistungen und Korrekturvorschläge beschränken. Dementsprechend wurden in vier weiteren, ganztägigen Workshops von Benutzervertretern und Entwicklern Vorschläge der Benutzer zum Maskendesign, Konzepte für Erfassungsabläufe und das Gesamtsystem (Informationsabfrage, Erfassung) diskutiert und mit den Vorstellungen der Entwickler abgestimmt; in der Regel wurden die Ergebnisse dieser Arbeitstage von den Benutzervertretern an ihrer Arbeitsstelle im Kollegenkreis besprochen und allfällige Korrekturvorschläge bzw. anders gelagerte Vorstellungen an die Projektgruppe weitergeleitet, die sich ihrerseits mit den Konsequenzen für das weitere Vorgehen, in Angriff zu nehmende Aufgaben bzw. deren Delegation an eine Arbeitsgruppe (z.B. für die Formulargestaltung und für Druckoutput), befasste.

Nachdem schliesslich bezüglich Grob- und zum Teil auch Detailspezifikation des Systems zwischen allen Beteiligten grosse Übereinstimmung herrschte, konnte sich die Projektgruppe vermehrt der Programmierung zuwenden. Damit waren auch wesentliche Aufgaben der Benutzergruppe abgeschlossen; ein Teil der Benutzervertreter arbeitete allerdings noch bei den Programmtests mit und übernahm auch Ausbildungsaufgaben. Zirka drei Jahre nach Projektbeginn wurde das System dem produktiven Einsatz übergeben und erfüllt – zur Zufriedenheit der Benutzer – seine Unterstützungsfunktion zuverlässig; lediglich einzelne Details waren noch zu korrigieren.

5 FAZIT

Neben generellen Problemen wie Zeit- und Termindruck waren in diesem Projekt aber auch noch andere Probleme aufgetreten. So bezogen sich die Benutzeranforderungen im allgemeinen auf praktische Details, waren z.T. auch widersprüchlich und erschwerten damit der Projektgruppe eine inhaltliche Clusterung, eine Schwerpunktsetzung und die Überführung in Konzepte.

Die Benutzergruppe war zwar repräsentativ für die verschiedenen Anwendungsbereiche, aber nicht für verschiedene Benutzer-'Typen' (sporadischer, täglicher Benutzer; Manager, Sekretärin). Eine Erweiterung der Gruppe hätte aber den Kleingruppenrahmen zerstört (die Anzahl wäre auf über 20 Personen gestiegen) und auch Qualifikationsprobleme nach sich gezogen. Die Projektgruppe war deshalb angesichts dieser Nachteile und in Kenntnis der jetzigen Vertreter der Ansicht, dass die bestehende Zusam-

mensetzung die Berücksichtigung unterschiedlicher Bedürfnisse gewährleistete.

Die Benutzervertreter wurden für ihre Mitarbeit im Projekt nicht explizit – z.B. zu einem gewissen prozentualen Teil der Arbeitszeit – freigestellt; der Einbezug in die Softwareentwicklung war für sie eine Nebenbeschäftigung mit entsprechenden Folgen in belastungsmässiger Hinsicht. Ausserdem wurde der Projektgruppe von den betroffenen Vorgesetzten eine zeitliche Beanspruchung der Benutzervertreter von sieben Tagen zugestanden (die aber weit überschritten wurden!); offenbar waren sie nicht gewillt, einen Teil ihres Mitarbeiterpotentials übergeordneten Organisationszielen zu 'opfern'. Als weiteres Problem erwies sich eine – wenn auch nicht übermässige – (von Vorgesetzten manchmal angekündigt: "Wir schicken nächstes Mal einen anderen") – Fluktuation in der Benutzergruppe; eine Benutzergruppe sollte vom Beginn bis Projektende mehr oder weniger in ihrer Zusammensetzung bestehen bleiben, um die Konstanz der Arbeit nicht zu gefährden.

Ein wesentliches Problem der Entwicklung bestand in der Realisierung der Anpassungsfähigkeit des Systems an die unterschiedliche Ablauforganisation in den Niederlassungen (v.a. unterschiedliche Grade an Dezentralisierung von Aufgaben). Die Niederlassungen hätten sich einer Standardisierung von organisationalen Abläufen vehement widersetzt; deshalb wurde nach einer Lösung gesucht, die mittels Parametereinstellung am System eine flexible, das heisst, stufenartige Dezentralisierung ermöglicht und damit den Organisationseinheiten genügend Spielraum zur Gestaltung 'eigener' Organisationsformen lässt.

Das bezüglich Benutzerbeteiligung wohl interessanteste Problem zeigte sich an der ersten Benutzertagung: Den Benutzervertretern wurde – nach der Vermittlung von Grobkonzepten – der Datenkatalog für jedes Bild und anschliessend die Dialogstruktur (mittels Matrix- und Baumdarstellung) präsentiert; diese isolierte Betrachtung der beiden Teilaspekte konnte von den Teilnehmern problemlos nachvollzogen werden. Für den Benutzer ist aber das Wissen entscheidend, an welchem Ort im Dialog er welche Informationen zur Verfügung hat; die verlangte gedankliche Integration von Daten und Dialog aufgrund von Papierunterlagen erwies sich an der Tagung als zu komplexe Aufgabe für die Teilnehmer. Die Benutzervertreter mussten 'zu Hause' die Abbildung der Baumstruktur des Dialoges Schritt für Schritt durchgehen und bei jedem Kästchen der Abbildung das entsprechende Blatt mit dem Datenkatalog beiziehen. Dies wurde von ihnen zwar 'brav' ge-

schluckt, in der Tagungsbeurteilung – die sehr positiv ausfiel – wurde allerdings einhellig der Wunsch geäussert, die Daten und Dialoge auf dem Bildschirm sehen und damit experimentieren zu können, um sich ein anschauliches, realistisches Bild zu machen. Mit einem Dialog- und Maskengenerator wäre die Erstellung eines derartigen Prototyps in einem halben Tag möglich gewesen.

Der Projektleiter – ein qualifizierter Analytiker mit zirka zehn Jahren Erfahrung in Softwareentwicklung – hatte anfänglich grosse Bedenken bezüglich der Beteiligung von Benutzern; seine Argumente waren z.B.: "Ich kenne die Benutzer und ihre Bedürfnisse selbst. Die Benutzer wissen ja nicht, was sie wollen, oder können es nicht verständlich sagen. Die Benutzer sind sich selbst uneinig, das bringt ja nichts ausser Zeitverzögerung und Probleme. Man kann ja sowieso nicht alle einbeziehen." Im Laufe der Zusammenarbeit sind diese Bedenken aber einer sehr positiven Einstellung – " Das ist sehr wichtig. Die wissen vieles besser als wir" – gewichen. Man könnte sagen, er hatte Spass an der Zusammenarbeit mit der Benutzergruppe bekommen.

Im beschriebenen Fall ist die Beteiligung von Benutzern bei der Softwareentwicklung weder in Projekthandbüchern noch in sonstigen Reglementen festgelegt. Fortschrittliche Projektleiter führen Benutzerbeteiligung jedoch aus eigener Initiative durch.

Fall 2: Bankabteilung KBT

1 AUSGANGSLAGE

Aufgabe der Abteilung KBT (Konzernbetreuung) einer Bank ist die umfassende Betreuung und Koordination aller Geschäftsbeziehungen, die Konzerne und ihre Tochterfirmen mit allen in- und ausländischen Geschäftsstellen der Bank unterhalten. Die Abteilung ist in Gruppen von zwei bis fünf Angestellten aufgeteilt, die jeweils eine Gruppe von Konzernen betreuen. Wichtigste Grundlage einer erfolgreichen Betreuung ist die umfassende Information über den Kunden, seine Vorhaben und Bedürfnisse; dazu müssen über verschiedene Quellen (von Datenbasen bis zu Zeitungen) Informationen eingeholt und überprüft werden. Auf operativer Ebene müssen z.B. für die Vergabe von Krediten komplexe Modellrechnungen durchgeführt und Offerten erstellt werden. Dementsprechend reicht das Spektrum der Aufgaben von relativ unstrukturierten Teilaufgaben (z.B. Informationsgewinnung) bis zu präzise definierten administrativen Abläufen.

Eine Gruppe besteht in der Regel aus einem Chef, einem oder mehreren Assistenten, Sachbearbeitern und einer Sekretärin. Der Chef bemüht sich vor allem um den direkten Kontakt mit den Kunden und trifft die wesentlichen Entscheidungen, die von den Assistenten vorbereitet werden; sie sammeln die nötigen Informationen, prüfen sie und führen z.B. Modellrechnungen durch. Die Sekretärinnen sind hauptsächlich mit der Eingabe von Texten und Daten, der Anfertigung von Grafiken und der Telefonbetreuung beschäftigt.

Zur Bewältigung dieser Aufgaben stand den ca. 30 Mitarbeitern der Abteilung ein umfangreiches Arsenal technischer Arbeitsmittel als Unterstützung zur Verfügung: Bildschirmterminals zum Grosscomputer, PCs, Textverarbeitungsautomaten, Telex, Fax, programmierbare Taschenrechner usw. Diese Vielfalt von Einzelgeräten bewirkte aber auch viele Doppelspurigkeiten und Medienbrüche; so mussten z.B. gleiche Informationen, Formulare mehrfach parallel in verschiedenen Systemen gespeichert und bei Änderungen nachgeführt werden. Aufwendige administrative Prozeduren und das wachsende Geschäftsvolumen führten schliesslich zu einer permanenten Überlastung der Sachbearbeiter. Zur genaueren Analyse dieses unbefriedigenden Zustandes und seiner Ursachen wurde eine Vorstudie durchgeführt; ein Organisator und ein Informatiker studierten sechs Wochen lang

die bankfachlichen Aufgaben und Arbeitsabläufe in dieser Abteilung. Anschliessend wurden in einem Workshop mit Vertretern der Abteilung deren Probleme und Bedürfnisse nach technischer Unterstützung diskutiert.

Schliesslich wurde beschlossen, im Rahmen eines Pilotprojektes ein System zu entwickeln, welches das Management in den Geschäftsbeziehungen und bei der operativen Abwicklung von einzelnen Geschäftsfällen unterstützt. Dabei sollten "sämtliche EDV-unterstützbaren Funktionalitäten der Konzernbetreuung in einem homogenen System zusammengeführt werden". Das Vorhaben kann als komplex und innovativ bezeichnet werden; sowohl die Menge als auch die unterschiedliche Struktur der pro Kunde zu verarbeitenden Informationen wie auch deren komplexe Vernetzung stellten besondere Ansprüche. Für die Gestaltung der Benutzungsoberfläche wirkte der heterogene Benutzerkreis mit unterschiedlichen Aufgaben, Bedürfnissen und stark variierender Benutzungshäufigkeit erschwerend. Die infrastrukturellen Voraussetzungen für die Durchführung dieses Projektes können als sehr gut bezeichnet werden, da z.B. modernste Entwicklungstools verfügbar waren. Ausserdem mussten die Projektverantwortlichen die Entwicklung *nicht* nach dem betriebsüblichen Vorgehen (starres Phasenmodell) realisieren; vielmehr sollte eine neuartige Vorgehensweise (iterativ-zyklische Entwicklung unter Einsatz von Prototyping) erprobt werden. Der Komplexität des Vorhabens entsprechend sollte die Entwicklung unter Beteiligung der zukünftigen Benutzer erfolgen.

2 PROJEKTORGANISATION

Die Realisierung des Projektes wurde einer Projektgruppe übertragen, die sich aus vier Informatikern, zwei ständigen Benutzervertretern und einem Softwareergonomen der Bank zusammensetzte. Die Arbeit in dieser Gruppe war durch eine netzwerkartige Kooperation, häufige gemeinsame Diskussionen und Entscheidungsprozesse sowie durch vollständige, definierte Aufgabengebiete der einzelnen Mitglieder charakterisiert. Die Gruppe bzw. deren formeller Leiter musste einem aus Vertretern der EDV und der Fachabteilung gemischt zusammengesetzten Ausschuss Bericht erstatten.

Aufgaben der Benutzervertreter waren:

(1) Informationen der Projektgruppe (Stand der Arbeit, Vorschläge usw.) an die anderen zukünftigen Benutzer zu vermitteln und Feedback einzuholen, das der Projektgruppe für weitere Arbeiten diente,

(2) aktive Mitarbeit bei der Arbeitsanalyse, der Konzeption der Funktionsteilung sowie der Benutzungsoberfläche,

(3) Qualifizierung der Benutzer.

Ihre Mitarbeit beschränkte sich also nicht nur auf das Abgeben, Einholen von Informationen sowie die Beurteilung von Vorschlägen, sondern erstreckte sich auf aktive Konzeptions- und Entwurfsarbeit. Dies bedingte allerdings ihre zeitliche Freistellung für diese Aktivitäten; vor allem in den Anfangsphasen des Projektes (ca. ein Jahr!) arbeiteten sie zu 100% ihrer Arbeitszeit für das Projekt.

Neben diesen ständigen Benutzervertretern wurden fünf weitere Benutzer im Sinne von 'Reviewern' jeweils nach Bedarf zur Beurteilung von Konzepten und Prototypen, für Tests und zur Beantwortung, Diskussion offener Fragen der Projektgruppe beigezogen. Die übrigen Benutzer wurden an verschiedenen Informationsveranstaltungen mit Diskussionsmöglichkeit über den Stand des Projekts informiert und ihre Standpunkte dazu erfragt. Mit diesem Beteiligungsmodell sollte ein repräsentativer Benutzereinfluss gewährleistet und auch Effizienzkriterien bzw. zeitlichen Beschränkungen der Benutzer für die Mitarbeit Rechnung getragen werden. Die grundlegende 'Philosophie' der Projektgruppe – kennzeichnend für das gesamte Vorgehen – lautete: "Wir haben den Benutzern Dienstleistungen zu erbringen!"

3 PROJEKTABLAUF

In einer ersten Informationsveranstaltung wurden sämtliche Mitarbeiter der Abteilung über das geplante Projekt und seine Ziele, mögliche Vorgehensweisen und die angestrebte Benutzerbeteiligung in Grundzügen informiert und Fragen der Teilnehmer beantwortet. Das Projekt begann aber eigentlich damit, dass die Entwickler sich durch intensives Studium bankfachlicher Literatur, von Stellenbeschreibungen, Reglementen usw. mit den Aufgaben der Benutzer vertraut machten. Dies bildete auch eine gute Grundlage für die anschliessend mit einem Grossteil der Benutzer geführten Gespräche über ihre konkreten Aufgaben, Probleme und technischen Unterstützungsbedürfnisse. Auf dieser Basis skizzierten die Entwickler auf Papier einzelne Szenarien im Sinne einer ersten Diskussionsbasis und erstellten zu Demonstrationszwecken einen Prototyp (ohne Funktionalität!) für einen Teil eines Arbeitsablaufes. Vertieft wurde die Zusammenarbeit in vier halbtägigen Workshops mit jeweils etwa sieben Benutzern; bei dieser Gelegenheit wurde auch der Prototyp vorgestellt und diskutiert. Die schnelle Umset-

zung erster Ideen in ein anschauliches Verhandlungsobjekt zeigte bei den Benutzern enorme Motivationswirkung für die weitere Mitarbeit im Projekt!

Ausgehend von den Ergebnissen der im Abschnitt 1 erwähnten Vorstudie, den Gesprächen mit den Benutzern und den Workshops wurden die allgemeinen Zielvorstellungen für das System präzisiert. Das System sollte die Benutzer durch eine Kombination von Daten und – für die bankfachlichen Aufgaben – massgeschneiderten Bearbeitungsinstrumenten (Funktionen) unterstützen; die Ziele können im einzelnen wie folgt benannt werden:

(1) Entlastung der Benutzer von administrativen Vorgängen durch Automatisierung,

(2) Unterstützung bei der Suche, Selektion, Aufbereitung, Vernetzung und Verdichtung von Informationen,

(3) Unterstützung bei der Bearbeitung von Fällen durch integrierte Instrumente (Teilautomatisierung),

(4) Unterstützung bei der Präsentation von Informationen (Grafik).

Aus der Sicht der Entwickler betrachtet, sollte sich das System an einem erweiterten Dokumentenbegriff orientieren. "Ein Dokument wird aber nicht nur als eine statische Darstellung von Informationen aufgefasst. Vielmehr werden den Dokumenten zwei weitergehende Funktionalitäten zugeordnet:

- Zum einen eine dynamische Verarbeitungsleistung, die den Zugriff auf Daten (aus Datenbanken und anderen Dokumenten) und deren Prozessierung im Dokument selber erlaubt.
- Zum anderen ein Referenzierungsmechanismus, der die sinnvolle Vernetzung der Dokumente untereinander und damit ein schnelles Finden von zusammenhängenden – strukturierten und unstrukturierten – Informationen ermöglicht."

In iterativen Entwicklungszyklen wurden weitere Prototypen erstellt und in Tests von den Benutzern auf Benutzbarkeit und Funktionalität evaluiert. Die Resultate dieser Tests flossen jeweils in die nächste Entwicklungsstufe ein. Die wichtigsten Stationen dieser Entwicklung werden nachfolgend skizziert:

(1) Innerhalb von zwei Wochen wurde mit einem Hypertextsystem ein Prototyp (ohne Funktionalität) erstellt, der hauptsächlich der "Modellierung des Grundbildes der Desktop-Benutzungsoberfläche, der Ausarbeitung von Beispieldokumenten und Masken und insbesondere für das Auf-

zeigen der Navigationsmöglichkeiten im System (Wechsel zwischen verschiedenen Dokumenten und Masken)" diente. Dieser Prototyp bot eine gute Basis für Diskussionen mit den Benutzern über Gesamtzusammenhänge und Darstellungsaspekte. Parallel dazu wurden die Arbeitsabläufe genauer untersucht.

(2) Mit einem anderen, mächtigeren Hypertextsystem mit Programmiermöglichkeiten wurde ein weiterer Prototyp (mit Pseudofunktionalität) realisiert. Damit konnten schon relativ komplexe Dialogabläufe realistisch simuliert und Details der Benutzungsoberfläche dargestellt werden. Die Benutzer erhielten in verschiedenen Reviewsitzungen ein anschauliches Abbild des zukünftig möglichen Systems; sie konnten anhand dieses Prototyps schon weitgehend ein Urteil abgeben, inwieweit er bezüglich Darstellung, Dialog und Funktionalität ihren Anforderungen entspricht.

(3) Schliesslich wurden mit Werkzeugen der vierten Generation Prototypen mit echter Funktionalität erstellt, zum Produkt weiterentwickelt und teilweise in das definitive System integriert.

Neben der Verwendung von Prototypingwerkzeugen wurden für die Detailspezifikation auch strukturierte Analyse- und Spezifikationsmethoden (Datenflussanalyse, semantische Datenmodellierung) eingesetzt. Zusätzlich zu den beschriebenen Entwicklungsschritten musste auch noch die Einbettung des Systems in das Gesamtsystem realisiert werden, was vor allem eine Frage der technischen Schnittstellengestaltung bedeutete.

4 FAZIT

Der beschriebene Fall veranschaulicht eine Entwicklungsstrategie, die durch Prototyping eine schrittweise Präzisierung der komplexen Anforderungen und realistische Umsetzungsmöglichkeiten erzielt. Die rasche Erstellung von Prototypen schon am Anfang des Projektes erlaubte einen frühzeitigen Einbezug der Benutzer, was die Zusammenarbeit bei der Systementwicklung positiv beeinflusste; ausserdem konnten Missverständnisse in der Anforderungsdefinition und Konzeptformulierung vermieden bzw. frühzeitig genug geklärt werden. Im einzelnen ergaben sich durch den Einsatz von Prototyping folgende Vorteile:

(1) Die Kommunikation zwischen Entwicklern und Benutzern wurde durch das Vorliegen eines anschaulichen Diskussionsobjektes erleichtert.

(2) Die schnelle Realisierung von Vorschlägen der Benutzer wirkte motivierend für deren weiteres Engagement im Projekt; sie erkannten ihren Einfluss auf die Systemgestaltung.

(3) Softwareergonomische Gestaltungsaspekte konnten einfacher erprobt und überprüft werden.

(4) Den Entwicklern wurde die Strukturierung von Ideen sowie deren Umsetzung in verschiedene Lösungsvarianten erleichtert; sie hatten mehr Gestaltungsspielraum.

Von grosser Bedeutung für die Entwicklungsarbeit und die Kommunikation mit den Benutzern war auch das bankfachliche Wissen, das sich die Entwickler zu Projektbeginn angeeignet hatten. Ohne diese Kenntnisse hätte der Einsatz von Prototyping die geschilderten Vorteile kaum in vollem Umfang erbracht! Bevor man mit der Erstellung von Prototypen beginnt, muss man sich auch Klarheit darüber verschaffen, welche Fragen damit beantwortet bzw. welche Ziele damit verfolgt werden sollen. Prototyping erfordert ein gezieltes, hypothesengeleitetes Vorgehen; sonst besteht nämlich die Gefahr, dass es zur ziel- und planlosen Bastelei oder Spielerei ausartet! Ferner ist bei der Wahl der Werkzeuge auf geringen Aufwand für Einarbeitung und Erstellung von Prototypen zu achten, sonst kostet die Erstellung von Prototypen bald einmal soviel wie die Produktion des eigentlichen Systems. Gerade bei komplexen, innovativen Projekten dürften sich aber bei Beachtung dieser Gesichtspunkte die Vorteile von Prototyping in Zukunft vermehrt zeigen.

Fall 3: Bibliothek

1 AUSGANGSLAGE

Anfang der 80er Jahre beschloss die Leitung einer grösseren öffentlichen Bibliothek, den vorhandenen Mikrofichenkatalog durch einen computergestützten On-line-Katalog (OPAC; On-line Public Access Catalogue) zu ersetzen; der Katalog sollte über verschiedene öffentliche und private Netzwerke frei zugänglich sein und einen heterogenen Benutzerkreis (Studenten, Wissenschaftler, Bibliothekare von Instituten, Betrieben und anderen Bibliotheken, Private) bei ihren unterschiedlichen Aufgaben unterstützen. In diesem umfangreichen Projekt sollte also ein relativ komplexes, für damalige Begriffe innovatives System entwickelt werden.

Zur Realisierung des Vorhabens wurde eine Projektgruppe gebildet; sie bestand aus dem Leiter der EDV-Abteilung, der die Projektleitung übernahm sowie – je nach Phase und Bedarf – fünf bis neun Analytikern, Programmierern. Ausserdem wurden zwei Bibliothekare in die Entwicklungsarbeit einbezogen; ihre Aufgabe war es, die Benutzerbedürfnisse in die Entwicklung einzubringen und allgemein die Benutzerinteressen zu vertreten. Konkret bedeutete dies für sie die Erarbeitung von Vorschlägen zur Funktionalität des Systems sowie die Beurteilung von Konzepten der Entwickler bezüglich Benutzungsoberfläche und Funktionalität.

Einem Phasenmodell folgend entstand schliesslich durch die intensive Zusammenarbeit der Entwickler und der zwei Bibliothekare ein System, das mittels Telekommunikation Literaturrecherchen nach formalen und sachlichen Kriterien (inklusive verschiedener Hilfsfunktionen wie Verknüpfung von Begriffen, Fragmentsuche, Sortieren usw.), On-line-Bestellungen von Dokumenten sowie die Neuanmeldung eines Benutzers bzw. die Änderung von Benutzerdaten erlaubt. Sämtliche Funktionen (Recherchen, Bestellung, Ausleihe, Verwaltung der Benutzerdaten) können on line mittels Telekommunikation ablaufen und von den Benutzern kontrolliert werden.

2 UNTERSUCHUNG

Anlässlich des fünfjährigen Betriebes des On-line-Kataloges war es ein Anliegen der Bibliotheksleitung, auf dem Wege der empirischen Sozialforschung mehr über das Benutzerverhalten im Umgang mit diesem Informa-

tionssystem in Erfahrung zu bringen. Die Resultate einer Untersuchung sollten einerseits – im Sinne einer Momentaufnahme – die Beurteilung und Bewertung des Systems aus Benutzersicht wiedergeben, andererseits auch Ansätze für längerfristige Verbesserungsmöglichkeiten aufzeigen.

Mittels Interviews und einer Fragebogenerhebung bei 473 Benutzern wurde die Beurteilung des Systems, speziell der Benutzungsoberfläche, durch die Benutzer hinsichtlich verschiedener Aspekte (Maskendesign, Dialogsteuerung, Sprache, Abkürzungen, Fehlermeldungen usw.) erfasst. Ausserdem sollte die Untersuchung darüber Aufschluss geben, welche Abfragearten und Funktionen wie häufig benutzt werden und wie deren Nutzen und Aufgabenangemessenheit beurteilt wird.

Die Resultate der Untersuchung zeigten, dass ein grosser Teil der befragten Benutzer die Potentiale des Systems nicht nutzt, sondern sich auf einfache Abfragen beschränkt und mit dem Ergebnis der Recherchen – und dem System – sehr zufrieden ist. Offensichtlich ist es für die überwiegende Mehrheit der Benutzer möglich, unter Ausnutzung eines relativ geringen, aber einfach handhabbaren Teils der Gesamtfunktionalität in effizienter Weise zu den gewünschten Ergebnissen zu gelangen.

Allerdings werden die komplexeren Abfragemöglichkeiten als eher kompliziert beurteilt. Offenbar bietet das System den Benutzern gute Unterstützung für gezieltes Suchen, ist aber für relativ unstrukturierte Recherchen für viele 'Normalbenutzer' zu kompliziert in der Bedienung. Es stellt sich die Frage, ob die Gestaltung der komplexeren Abfragen zu sehr von bibliothekarischem Denken – das heisst, der Sicht von Spezialisten – mitgeprägt wurde. Hier macht sich die Schwierigkeit der Gestaltung eines öffentlichen Systems bemerkbar, das einem heterogenen Benutzerkreis mit entsprechend unterschiedlichen Aufgaben, Unterstützungsbedürfnissen und Vorkenntnissen sowie stark variierender Benutzungshäufigkeit Rechnung tragen sollte.

Um auch den 'Normalbenutzern' den Zugang zu den komplexeren Abfragen zu erleichtern, gibt es prinzipiell zwei Möglichkeiten:

(1) Verbesserung des Ausbildungsstandes (durch erweitertes Kursangebot, Abgabe einer Detailanleitung und eines 'Quick Reference Manual') und

(2) Änderung des Systems.

Verschiedene, mit vertretbarem Aufwand kurzfristig realisierbare Änderungen können hinsichtlich Maskengestaltung, Sprache, Abkürzungen, Dia-

logsteuerung usw. erfolgen; hierzu konnte den Kommentaren der Benutzer in den Fragebogen eine Fülle von Anregungen entnommen werden.

Mittel- und längerfristig ist allerdings an eine konzeptionelle Überarbeitung des Systems unter Berücksichtigung neuer Dialogtechniken und moderner Medien (wie z.B. CD-ROM) zu denken. Als eine erste Grundlage für diese Weiterentwicklung sollten die Ergebnisse eines Workshops von Benutzern und Entwicklern dienen.

3 WORKSHOP

Generelles Thema des Workshops waren Verbesserungs- bzw. Erweiterungsmöglichkeiten des Systems. Im einzelnen wurden folgende Ziele benannt:

(1) Hinweise, Vorschläge für eine Systemverbesserung erarbeiten,

(2) gegenseitiges Verständnis von Benutzern und Entwicklern fördern (Entwickler sehen, wie Benutzer mit dem System umgehen; Benutzer sehen technische Probleme der Entwickler),

(3) Kontakte der Benutzer sowie der Entwickler untereinander fördern.

Der Workshop sollte insgesamt einen halben Tag dauern und mit einem gemeinsamen Essen abgeschlossen werden. Teilnehmer des Workshops waren – neben den Entwicklern und einigen Bibliothekaren – 24 repräsentative, qualifizierte, kritische und interessierte Benutzer, die aufgrund ihrer Angaben im Fragebogen ausgewählt und eingeladen wurden (ca. 30% der befragten Benutzer hatten sich im Fragebogen bereit erklärt, an einem Workshop teilzunehmen!). Der Ablauf des Workshops gestaltete sich wie folgt:

(1) Einleitend wurden Ziel und Zweck der Veranstaltung sowie die Umfrageergebnisse vorgestellt.

(2) Anschliessend wurde in fünf themenspezifischen Arbeitsgruppen, deren Moderation je ein Softwareentwickler zusammen mit einem Bibliothekar übernahm, über das System diskutiert, anhand von Beispielen am Bildschirm wurden on line Probleme aufgezeigt und Verbesserungsvorschläge erarbeitet.

(3) Es folgte – im Plenum – die Ergebnispräsentation der Arbeitsgruppen. Vor dem gemeinsamen Essen ergab sich dann auch eine Diskussion über Möglichkeiten einer weiteren Kooperation.

Der Workshop wurde von allen Teilnehmern als erfolgreicher Beginn einer verstärkten Zusammenarbeit gewertet; die gesetzten Ziele wurden erreicht:

(1) Es ergab sich wiederum eine ganze Reihe von Verbesserungsvorschlägen; in einer gemeinsamen Sitzung der Moderatoren der Arbeitsgruppen wurden diese Vorschläge aufgearbeitet und einige davon auch sofort umgesetzt, was von den Benutzern mit Freuden festgestellt wurde.

(2) Die Benutzer erkannten dadurch, dass der Workshop keine Alibiübung war, sondern dass ihre Beiträge ernst genommen wurden. Ausserdem erhielten die Teilnehmer eine Zusammenfassung der aufgearbeiteten Workshopergebnisse (Feedback!).

(3) Ferner wurde auch das gegenseitige Verständnis von Entwicklern und Benutzern gefördert, gab es doch auf beiden Seiten einige 'Aha'-Erlebnisse zu verzeichnen.

So ist es jetzt z.B. den Benutzern verständlich, dass bei der bestehenden Datenbank das Rückwärtsblättern in Maskenfolgen nur mit immens hohem Programmieraufwand realisierbar ist; die Entwickler haben z.T. mit Erstaunen zur Kenntnis genommen, dass die Gedankengänge der Benutzer bei Recherchen im System nicht immer mit den von ihnen realisierten Dialogabläufen übereinstimmen. Schliesslich haben die Benutzer untereinander ihre vielfältigen und z.T. widersprüchlichen Anforderungen kennengelernt. Alle Teilnehmer hatten das Bedürfnis nach weiterer Zusammenarbeit; als Möglichkeiten – die sich nicht ausschliessen, sondern kombinierbar sind – wurden die Einrichtung einer Mailbox, die Bildung einer User-Group, die Ernennung eines Benutzerkoordinators sowie weitere Workshops genannt.

Zur Zeit wird ein zweiter Workshop geplant; ausserdem befindet sich die Etablierung einer gemischt zusammengesetzten Gruppe im Diskussionsstadium (Softwareentwickler, Bibliothekare und weitere Benutzer). Aufgaben dieser Gruppe wären die Durchführung weiterer Aktivitäten mit Benutzern sowie – darauf aufbauend – die Erarbeitung von Konzepten für ein zukünftiges System bis zur Erstellung von Prototypen.

Dieses Fallbeispiel zeigt, wie die – gerade bei einem öffentlichen System mit grösstenteils unbekannten Benutzern – traditionellerweise vorhandene Distanz zwischen Entwicklern und Benutzern durch eine Umfrage und einen Workshop überbrückt und die weitere Entwicklung in eine kooperative Form überführt werden kann.

Fall 4: Standardsoftware bei der ADI

1 AUSGANGSLAGE

Das diesem Fallbeispiel zugrundeliegende, gemeinsam zwischen dem Institut für Arbeitspsychologie der ETH Zürich und der ADI Software, Karlsruhe, durchgeführte Forschungsprojekt BOSS (Benutzerorientierte Softwareentwicklung und Schnittstellengestaltung) hatte die Aufgabe, vorhandene Kriterien und Methoden zu sammeln, zu systematisieren, weiterzuentwickeln sowie Benutzungstests zu unterziehen. In der Zusammenarbeit von Arbeits- und Organisationspsychologen und dem Softwarehaus (ADI) wurden die arbeitswissenschaftlichen Vorgaben aufbereitet und in Realisierungsvarianten eines konkreten Produktes (Datenbanksystem ADIMENS) erprobt. Die vorgegebenen Kriterien wurden in eine Relation zu den hard- und softwaretechnischen Möglichkeiten gesetzt und unter Mitwirkung der Endanwender sukzessive in der Produktentwicklung berücksichtigt. Die erstellten Produktvarianten wurden dabei im arbeitspsychologischen Labor ausführlichen Tests hinsichtlich der Erreichung der angestrebten Ziele (z.B. benutzerorientiertes Gestaltungskonzept) unterzogen und entsprechend verbessert. Aufgrund der weiten Verbreitung des Produktes (ca. 35'000 vollgrafische Installationen im beruflichen und privaten Einsatzkontext, insgesamt ca. 85'000 Installationen von Personalcomputern bis zu UNIX Workstations im Markt) wurde auch die Möglichkeit genutzt, einen grösseren Benutzerkreis in diesen Prozess einzubeziehen. Parallel zur Produktentwicklung untersuchte das BOSS-Projekt typische Softwareentwicklungsprozesse im Hinblick auf die Benutzerpartizipation bei der ADI.

Im folgenden werden erste betriebliche Erfahrungen und eingeschlagene methodische Wege im Kontext der benutzerorientierten Standardsoftwareentwicklung des vollgrafischen Datenbanksystems ADIMENS aufgezeigt. Das Datenbanksystem ADIMENS eignet sich aus drei Gründen besonders als dynamische Basis für die Benutzerpartizipation bei der Dialoggestaltung:

(1) *Adimens* bietet *zwei* verschiedene Benutzungsoberflächen auf *einer* einzigen Anwendungskomponente: ASCII = menüorientiert mit Funktionstasten (CUI = 'character oriented user interface'), DESKTOP = desktoporientiert unter direktmanipulativer Nutzung von Maus und Ikonen (GUI = 'graphic oriented user interface').

(2) Für experimentelle Untersuchungen stehen wegen der hohen Anzahl von Installationen Anfänger, Fortgeschrittene und Experten als Testbenutzer zur Verfügung (siehe Benutzungstests im Teil B, Kapitel 4.14).

(3) Der dem *Markenartikelmarkt* vergleichbare Markt der Standardsoftware auf Personalcomputern, zu dem *Adimens* gehört, weist *Charakteristika* auf, die für *Benutzerpartizipation* geeignet sind:
• Die *Lebens-Versions-Zyklen* der Software werden einhergehend mit dem Preis-Leistungs-Rennen auf der Hardwareseite ständig kürzer.
• Softwareanbieter treffen zunehmend auf *Neueinsteiger* in der PC-Welt, die einerseits die Qualität der gekauften Ware ähnlich beurteilen wie bei Artikeln des Alltagsbedarfs, andererseits *aus der anonymen Nach-dem-Kauf-Situation herausbrechen wollen* mit dem Wunsch nach persönlicher Beratung und Kontaktaufnahme.
• Langzeitbenutzer geben neben kritischen Hinweisen, auch ohne grössere Aufforderung, konkrete Vorschläge zur Implementierung für sie wichtiger Funktionen. Andererseits ermutigt die hohe 'Update-Treue' einen Softwarehersteller, benutzerseitig eingebrachte Verbesserungen bez. Handhabung und Funktionalität als neue Versionen auf den Markt zu bringen.

Schliesslich kann die bereits beim Entstehungsprozess stattfindende Benutzer-Entwickler-Kommunikation eine kulturell notwendige Anpassung der Programme bewirken. Dies ist allerdings mit einem Aufwand verbunden, der von US-amerikanischen Unternehmen nicht geleistet werden will und somit von lokalen Unternehmen übernommen werden muss.

2 PROJEKTMANAGEMENT UND PROJEKTABLAUF

Zu Projektbeginn lag ADIMENS in seiner zweiten Generation vor (Version 2.*). Die vollgrafische DESKTOP-Variante (Version 2.1) und die ASCII-Variante (Version 2.23) wurden, basierend auf einem gemeinsamen Datenbankkern (AdiPROG), organisatorisch getrennt entwickelt. Als Partizipationsangebote dienten in eher passiver Weise Kundenkontakte im Support, aber auch bereits in der Öffentlichkeit verkaufsfördernde Veranstaltungen (Messen, Roadshows, Händlertage, Demonstrationen, Schulungen bei einzelnen Kunden). Der empirische Vergleich beider Varianten lenkte jedoch – infolge seines überraschenden Ergebnisses hinsichtlich der Lernförderlichkeit und Überlegenheit der vollgrafischen DESKTOP-Variante auch bei Experten – die Konzentration auf die Weiterentwicklung dieser DESKTOP-Variante. Die gleichzeitig eingebrachten Gestaltungshinweise und funktio-

nellen Anforderungen erforderten in der Entwicklungsorganisation der ADI datenbankseitig die Bereitstellung eines neuen Datenbankkernes sowie die grundlegende Auseinandersetzung mit den existierenden bzw. sich am Markt abzeichnenden Oberflächensystemen, um Einschränkungen der benutzten DESKTOP-Variante langfristig zu überwinden. Daneben wurden die parallelen Weiterentwicklungen synchronisiert. Gleichzeitig wurde das Partizipationsangebot aktiver und systematischer:

(1) Bei Vor-Ort-Aktivitäten (Messen, Händlertag) wurden Benutzerzufriedenheit und Gestaltungshinweise im Kundengespräch akzentuiert.

(2) In die existierende Fehlerdatenbank für die Produktentwicklung wurden für die Weiterentwicklung auch Benutzerwünsche aufgenommen.

(3) Die Sprechstundenzeit in der Kundenbetreuung (Hotline) wurde erhöht und mit Mitarbeitern aus Vertrieb, Dokumentation und Entwicklung besetzt.

(4) Ein Benutzertreffen mit ausgewählten Langzeitanwendern wurde veranstaltet.

(5) Gemeinsam mit einer Fachzeitschrift wurde eine Fragebogenaktion durchgeführt, um das Wissen über die Benutzer (Benutzerprofil) zu vertiefen.

(6) Zu ausgewählten Aufgabenstellungen wurden induktive und deduktive Benutzungstests durchgeführt; diese Benutzungstests dienten einerseits zur Überprüfung der getroffenen Gestaltungsmassnahmen und andererseits zur Gewinnung neuer Gestaltungsvorschläge (siehe Abbildung 16).

Während die Massnahmen (1) bis (4) vor allem Verbesserungswünsche bez. Funktionalität und Handhabbarkeit lieferten, zeigte (5) darüber hinaus sowohl durch die starke Resonanz (220 Rücksendungen) als auch durch die inhaltliche Detaillierung (z.T. über 100 Seiten Begleittext und -skizzen) eine hohe Partizipationsbereitschaft. Allerdings liessen gerade (4) und (5) wegen des teilweise erheblich unterschiedlichen Anwendungskontextes auch Probleme der Singularität der gewonnenen Hinweise erkennen. Mit Hilfe der Anregungen der Benutzer wurde ein Jahr nach der Vorstellung der Version 2.3 die Version 3.0 im Herbst 1989 im Markt eingeführt.

Mit ADIMENS 3.0 erhielten die Benutzer wesentliche Pluspunkte für ein aufgabenangemesseneres Arbeiten (z.B. Interaktiver Join, Arbeitsumgebung, Multiples Sorting). Die Auswirkungen der umgesetzten Gestaltungs-

vorgaben wurden in entsprechenden Benutzungstests auf ihre Gebrauchstauglichkeit überprüft.

Die positiven Erfahrungen bei der Entwicklung führten organisatorisch bei der ADI zu einer stärker synchronisierten Parallelisierung der Projektgruppen unter der 'Patenschaft' je eines Produktverantwortlichen aus Vertrieb und Entwicklung mit frühzeitiger Einbeziehung der Dokumentation und interner Benutzer, z.B. aus dem Aufgabenfeld der Kundenbetreuung. Bei der Weiterentwicklung der Produkte werden seither Beta-Tests mit externen Benutzern durchgeführt, die auch die Benutzbarkeit berücksichtigen.

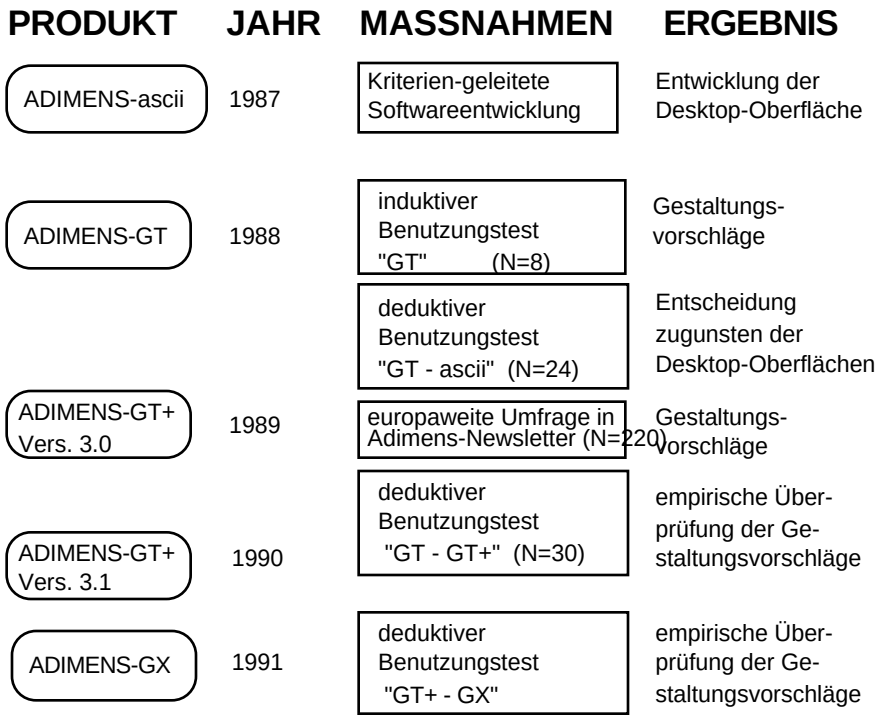


Abbildung 16: Übersicht über den Verlauf der durchgeführten Benutzungstests und Umfragen im Rahmen der Standardsoftwareentwicklung von ADIMENS in den Jahren 1988 bis 1991.

Als neues partizipatives Instrument werden seither jedem Produkt Registrierkarten beigelegt, deren Kurzfragebogen auf der Rückseite wichtige

Hinweise zum Benutzerprofil, Anwendungskontext und Raum für Kritik bietet. Die Ergebnisse der regelmässigen Auswertung dieser Informationen beeinflussten in erheblichem Mass Leistungsmerkmale und Handhabbarkeit der – wiederum nach einem Jahr – fertiggestellten Version 3.1 (Bildverwaltung, Query-by-Example, Datentyp Zeit, u.v.m.).

Parallel zur Version 3.1 entstand dann eine ADIMENS-Variante unter Windows 3.0, dessen grafische und Multitaskingmöglichkeiten die Implementierung weiterer grundlegender Gestaltungsvorgaben zulassen (Öffnen mehrerer Datenbanken, Multiwindowing auch bei Masken, direktmanipulativer Datenaustausch auch zwischen unterschiedlichen Anwendungsprogrammen). Zur Umsetzung der in der zweiten und dritten Generation benutzerseitig geforderten Leistungsmerkmale (z.B. data dictionary) wird der oben erwähnte neue Datenbankkern für die Anwendungskomponente verwendet.

3 FAZIT

Die gemachten Erfahrungen können im Hinblick auf eine benutzerorientierte Softwareentwicklung im Bereich der Standardsoftware aus der Sicht des entwickelnden Softwarehauses zu drei miteinander in Wechselwirkung stehenden Anforderungen zusammengefasst werden:

(1) In der *Aufbauorganisation des Softwarehauses* sind Massnahmen zu treffen, die über die marktnotwendige Kundenorientierung und damit verbundene verkaufsfördernde Strategien zum Absatz der entwickelten Produkte hinausgehen in bezug auf:

   (a) die Förderung der Orientierung am Benutzer in allen Unternehmensbereichen durch internen exemplarischen Einsatz von Produkten (inkl. Varianten) so früh wie möglich (Beta-Test), so breit wie möglich (Marketingdatenbank, Vertriebssteuerung und Auftragsabwicklung, Produktdatenbank), verbunden mit regelmässigen internen Präsentationen bereits im frühen Entwicklungsstadium (aktuelle Auseinandersetzung mit repräsentativen Benutzern in unmittelbarer Nähe sowie Anreicherung des Pflichtenheftes mit aktuellen Verbesserungsmöglichkeiten, z.B. aus der Kundenbetreuung);

   (b) die Auffächerung des Kundenkontaktes im Unternehmen sowie auf externen Veranstaltungen unter direkter Beteiligung möglichst vieler Mitarbeiter;

(c) die Verfügbarkeit einer Ablauforganisation, die Resultate aus den mittels (a) und (b) geöffneten Kanälen der Benutzerbeteiligung in neue Produkte oder Dienstleistungen (Training, Workshops) einfliessen lässt.

(2) Benutzerorientierte Softwareentwicklung benötigt für ihre konsequente Umsetzung ein Qualitätssicherungssystem, das:

(a) die Produktqualität auf Arbeitsqualität zurückführt und jeden Mitarbeiter durch seine verantwortungsbewusste Arbeit an der Qualität derjenigen Produkte und ihrer Komponenten beteiligt, die sein Aufgabengebiet berühren;

(b) im gesamten Entstehungsprozess an den Stationen eines Qualitätskreises (nach DIN ISO 9004) Möglichkeiten der Benutzerpartizipation berücksichtigt;

(c) durch hinreichende Systematisierung die fortlaufende Verbesserung der Produkte mit dem Ziel einer menschengerechten Gestaltung von Software fördert.

(3) Als wesentliche Instrumente der hier erprobten Softwareentwicklungsmethode, die eine benutzerorientierte Softwareentwicklung im Detail gewährleisten, sind zu nennen:

(a) Benutzerprofil und -kontakte (von der Breite bis in die Tiefe): Registrierkarte mit Kurzfragebogen zum Benutzerprofil, Anwendungskontext und Raum für Kritik, Berichterstattung in Fachzeitschriften zur Aktivierung der Leser, Anwender, Vor-Ort-Aktivitäten (Messen, Handel), Telefonnotizen (Hotline) und Wunschzettelbriefe in der Kundenbetreuung, Diskussionen mit Benutzern (z.B. für die Lastenhefterstellung, telefonisch oder über Benutzertreffen);

(b) Mitarbeiterbeteiligung: produktverantwortliche 'Paten' aus Vertrieb und Entwicklung auch noch nach begonnener Vermarktung, Kundenbetreuung in personeller Mischbesetzung aus den Bereichen Vertrieb, Dokumentation, Entwicklung, Projektdurchführung parallelisiert in Gruppen aus den Bereichen Marketing, Vertrieb, Dokumentation, Entwicklung von Anfang an, Mitarbeiter des Unternehmens als repräsentative Benutzer;

(c) Produktqualität: Fehler-, Wunschdatenbank und Versionshistorie innerhalb eines Qualitätssicherungssystems zur Unterstützung bei

der Produktfestlegung und -entwicklung, Qualitätsprüfung sowie der Kundenbetreuung.

Die entstandenen Strukturen und Instrumente können neben Bewertungs- und Gestaltungsschemata ein Softwareengineering für eine menschengerechte Gestaltung von Software unterstützen. Die gemachten Erfahrungen zeigen deutlich, dass Innovationen unter einer ganzheitlichen Sichtweise erforderlich sind. Einer Sichtweise, die den Benutzer als autonomes, sich qualifizierendes Subjekt sieht. Dies steht im Einklang mit dem im Rahmen des Projektes entstandenen Integrationsmodell für die Gestaltung partizipativer Softwareentwicklungsprozesse.

Fall 5: Die Einführung von PPS

In diesem Fall werden die Ziele, das Vorgehen und die damit verbundenen Probleme bei der Einführung eines rechnergestützten Produktionsplanungs- und Steuerungssystems (PPS) in einem grösseren Unternehmen beschrieben. Das Beispiel verdeutlicht die besondere Bedeutung arbeitsorganisatorischer Aspekte bei der Einführung komplexer, integrierter Softwarelösungen.

Im zeitlichen Verlauf des Projektes sind drei wesentliche Hauptphasen zu unterscheiden: Phase I, geprägt von einer 'eher arbeitsorientierten Idee'; Phase II mit einer 'technikorientierten Realisierung'; Phase III mit der 'arbeitsorientierten Korrektur'.

1 PHASE I: DIE 'EHER ARBEITSORIENTIERTE IDEE'

Zur Reduktion von Lagerbeständen und Durchlaufzeiten sowie vor allem auch zur Erhöhung der Termintreue sollte im Rahmen eines Projektes in einem grösseren Unternehmen das alte, batchorientierte PPS-System durch ein neues, integriertes und interaktives PPS-System abgelöst werden. Für die Erreichung dieser Ziele wurde u.a. vorausgesetzt, dass die Abteilungen Entwicklung, Verkauf, Beschaffung, Disposition und Arbeitsvorbereitung organisatorisch stärker integriert werden; dazu sollte eine funktional integrierte Fertigungsadministration (Disposition) geschaffen werden, welche Aufgaben der Verkaufslogistik, Entwicklungsadministration, Arbeitsvorbereitung und Beschaffung (Termin- und Mengenverantwortlichkeit für Bestellaufträge) wahrnimmt. Die informationstechnische Vernetzung dieser Organisationsstruktur sollte durch das neue System ermöglicht werden. Die organisatorischen Ziele enthielten auch die Vorstellung, die Auftragsfreigabe sowie die Terminfeinplanung (über ca. eine Woche) in die Fertigung zu verlagern. Bis zu diesem Zeitpunkt wurde die Fertigung durch das 'Terminbüro' in sehr intransparenter Weise mit Aufträgen gespeist. Im Rahmen des Projektes sollte und wurde diese 'Instanz' daher abgeschafft.

Bereits vor der Entwicklung dieser technisch-organisatorischen Ideen war im Rahmen eines neuen Informatikkonzeptes ein integriertes PPS-Standardsystem – welches u.a. Funktionen zur Kundenauftragsverwaltung, Produktionsprogrammplanung, Material- und Kapazitätsbedarfsplanung, Materialwirtschaft, Arbeitsplan- und Stücklistenverwaltung, Fertigungsauftragsverwaltung und -freigabe sowie Lagerbewirtschaftung enthält und entsprechend die gesamte logistische Kette technisch unterstützt – durch die Informatikabteilung favorisiert und von der Geschäftsleitung zur Einfüh-

rung genehmigt worden. Die genannten organisatorischen Ideen des Projekts hatte der Leiter der Abteilung Informatik vor dem Hintergrund der Möglichkeiten dieses Systems entwickelt.

2 PHASE II: DIE 'TECHNIKORIENTIERTE REALISIERUNG'

Die eher arbeitsorientierte Idee des Projektes – mit der Einführung eines technisch integrierten PPS-Systems *auch* eine funktional stärker integrierte Organisationsstruktur zu schaffen – wurde im Rahmen des Projektes zunächst nicht realisiert. Im Anschluss an die formulierten Ideen folgte kein weiterer Projektschritt, in dem verschiedene technisch-organisatorische Varianten ausgearbeitet wurden. Uneinigkeit in der Geschäftsleitung und auch unter den Abteilungsleitern bezüglich der technisch-organisatorischen Erneuerung (Pro- und Contra-Lager!) war einer der zentralen Gründe dafür, weshalb man sich zunächst auf die Einführung des technischen Systems konzentrierte.

Dazu wurde zunächst eine 'Projektgruppe' eingerichtet, die sich aus zirka 30 Mitarbeitern der verschiedenen projektrelevanten Bereiche zusammensetzte. Die Mitarbeit erfolgte ohne spezifische Qualifizierung für die Projektarbeit sowie ohne zeitliche Freistellung neben dem Tagesgeschäft. Die Projektorganisation war durch eine unklare Führungsstruktur und Kompetenzregelung gekennzeichnet. Die genannten Merkmale führten zu einer Form der Projektarbeit, die durch Schwerfälligkeit und Ineffizienz gekennzeichnet war. Diese Projektgruppe wurde daher aufgelöst und durch eine neue Projektgruppe ersetzt.

Diese neue Projektgruppe setzte sich aus drei Vertretern der internen Informatik, fünf Fachbereichsvertretern (aus der Disposition, Arbeitsvorbereitung, Qualitätssicherung und Entwicklung) sowie – zeitweise – einem Berater des Softwarelieferanten zusammen. Als Projektleiter fungierte der Leiter der Informatikabteilung (vgl. Abbildung 17). Vom Projektausschuss und Projektleiter wurden als Fachbereichsvertreter bewusst junge Mitarbeiter mit nicht allzu langer Betriebszugehörigkeit und guten fachlichen Qualifikationen ausgewählt. Sie wurden für die Mitarbeit in der Projektgruppe vom Tagesgeschäft freigestellt. Hierarchisch waren die Fachbereichsvertreter auf Gruppen- und Abteilungsleiterebene angesiedelt oder wurden nach Abschluss des Projektes in solche Positionen befördert. Diese Fachbereichsvertreter wurden spezifisch in Fragen des Projektmanagements durch eine externe Institution wie auch vom Systemlieferanten systemspezifisch ausgebildet.

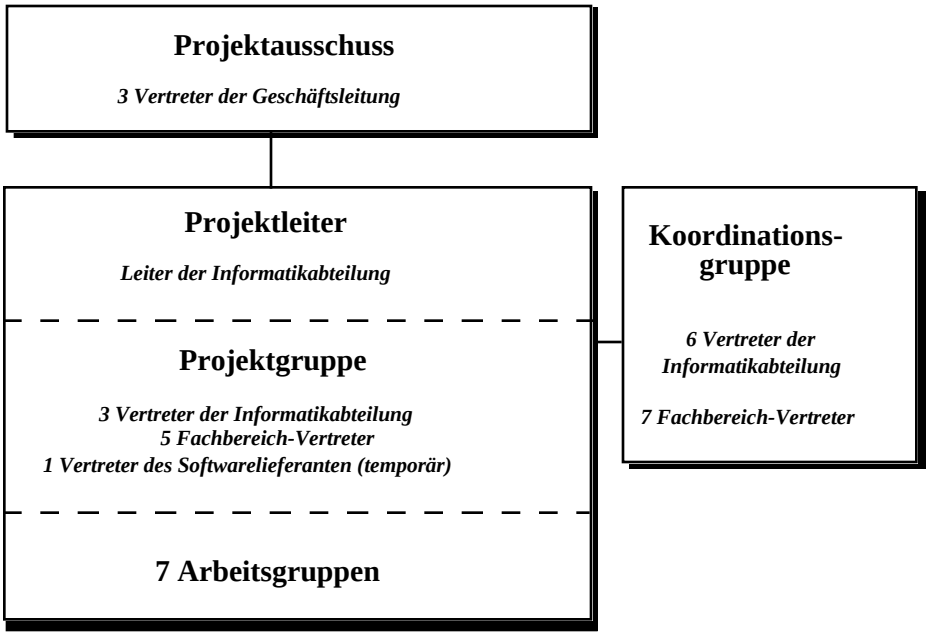


Abbildung 17: Die Aufbauorganisation des Projektes.

Die gesamte Projektgruppe wurde in bereichsspezifische Arbeitsgruppen mit je zwei bis drei Personen unterteilt. Die Verantwortung für die einzelnen Arbeitsbereiche lag jeweils bei einem Mitglied der Projektgruppe. Die einzelnen Mitglieder der Projektgruppe arbeiteten dabei in mehreren Arbeitsbereichen mit. Insgesamt sieben Arbeitsbereiche wie Bedarfsplanung, Bestandesführung, Materialklassifikation usw. wurden in diesem Sinne definiert. Zum Teil wurden auch ad hoc andere Mitarbeiter der Fachbereiche zu spezifischen Fragen beigezogen.

Die Projektgruppe wurde für die Dauer des Projektes in einem Grossraumbüro örtlich integriert, was sich als sehr kommunikationsförderlich erwies und zu einem guten Klima beitrug. Die gesamte Projektgruppe tagte wöchentlich. Die Ergebnisse der Arbeitsgruppen sowie der wöchentlichen Sitzungen wurden in Beschlussprotokollen – unterschiedlich gut und genau – dokumentiert und bei Bedarf durch den Projektleiter an den Projektausschuss weitergeleitet. Der Projektausschuss setzte sich aus drei Vertretern der Geschäftsleitung zusammen und tagte in sehr unregelmässigen Abständen (im Durchschnitt etwa alle zwei Monate).

Neben den bereits beschriebenen Gremien gab es zusätzlich eine Koordinationsgruppe, welcher ebenfalls der Projektleiter vorstand. Diese Gruppe setzte sich aus sieben Fachbereichsvertretern und sechs Vertretern der Informatikabteilung zusammen. In dieser Koordinationsgruppe wurden im Rahmen von zweiwöchigen Sitzungen Anpassungen und Schnittstellenfragen zu bereits implementierten Applikationen (wie z.B. zur Finanzbuchhaltung) geklärt.

Da die Wahl des technischen Systems ohne detaillierte innerbetriebliche Ist-Analyse und ohne weitere Konzeptspezifikation bestimmt worden war (das gewählte System teilte auch die Projektgruppe zeitweise in ein Pro- und ein Contra-Lager!), stand im weiteren Projektverlauf die Beantwortung folgender Fragen im Vordergrund: Was kann das System? Welche Situation ist in unserer Firma vorzufinden? Wie müssen wir uns verändern, damit wir systemkompatibel werden und trotzdem "unsere Organisation beibehalten können"?

Entsprechend wurden in der Folge die bestehenden Arbeitsabläufe und Informationsflüsse mittels daten- und funktionsorientierten Methoden analysiert und die 'notwendigen' systemkompatiblen Abläufe konzipiert. Die Arbeitsteilung zwischen und in den Abteilungen wurde dabei allerdings nicht mehr hinterfragt. Die systemkompatiblen Konzepte wurden in den verschiedenen Arbeitsgruppen spezifiziert. Im Rahmen dieser Arbeitsschritte wurden z.B. Stücklistenstrukturen festgelegt, Materialien zu Klassen zusammengefasst, Auftragsfreigabeverfahren definiert usw.

In dieser Situation bestand für die Projektgruppe ein grosses Problem in der Komplexität des integrierten PPS-Systems, das heisst, der vollständigen Ermittlung und Erfüllung aller systembedingten Anforderungen. Verschärfend wirkte der grosse Zeitdruck für die Realisierung dieser Projektschritte. Die programmtechnischen Systemanpassungen der Spezifikationen wurden durch Mitarbeiter der internen Informatikabteilung geleistet. Zu Simulations- und Testzwecken wurden Prototypen, die im Laufe des Projektes vom Systemlieferanten mit Echtdaten geliefert wurden sowie ein Testsystem mit der jeweils aktuellen Systemversion eingesetzt. Nach erfolgreichen Modul- und Systemtests wurden die Daten vom alten System vollständig auf das neue System überspielt. Das neue PPS-System wurde schliesslich 'über Nacht' in Betrieb genommen.

Die Mitarbeiter der verschiedenen Fachabteilungen wurden das erste Mal zirka neun Monate vor der Einführung durch die Hauszeitung über das Projekt informiert; im weiteren flossen die Informationen über 'informelle Kanäle'. Der grosse Zeitdruck während der Projektrealisierung war einer der Gründe, warum die Mitarbeiter nicht angemessen aufgeklärt, vorbereitet oder stärker in die Veränderung einbezogen wurden.

Als Folge machten sich zum Teil Desinteresse, Abneigung und auch Angst unter den Mitarbeitern gegenüber der technischen Neuerung bemerkbar; diese Problematik bekam die Projektgruppe u.a. an Bemerkungen wie "PPS wird das dann schon machen!" zu spüren.

Die Schulung der Mitarbeiter wurde ebenfalls von der Projektgruppe organisiert, dokumentiert und durchgeführt. Zunächst wurden ca. 150 Mitarbeiter geschult; die Schulung erfolgte z.T. vor, mit und nach der Einführung. Im Rahmen der Schulungseinheiten wurde Wissen über die Bedienung des Systems sowie über neue Arbeitsabläufe vermittelt.

Als grundsätzliche Probleme erwiesen sich bei der Realisierung der Ausbildung die mangelnden praktischen Erfahrungen der Projektgruppe wie auch zum Teil die Motivation und Akzeptanz der Mitarbeiter in bezug auf die Schulung. Nachher wurde auch eingesehen, dass man ein zu statisches Schulungskonzept entwickelt und realisiert hatte, das sich zuwenig an den Abläufen und den konkreten Arbeitstätigkeiten der Beschäftigten orientierte.

Der technische Teil des Projektes von der Projektidee bis zum Abschluss der Einführung des PPS-Systems wurde im budgetierten Rahmen mit einem Aufwand von zirka 20 Personenjahren und einer gesamten Dauer von ca. 2,5 Jahren realisiert. Für die eigentliche Projektrealisierung – das heisst, die konkrete Systemspezifikation, programmiertechnische Anpassung, Schulung und Einführung – stand jedoch ein viel geringerer Zeitraum zur Verfügung. Unter sehr starkem Termindruck wurden diese Projektschritte in ca. neun Monaten abgewickelt.

Die bisherigen Projektkosten belaufen sich auf zehn Millionen Franken. Der Hauptanteil fällt dabei auf Personalkosten für freigestellte Projektmitarbeiter der Fachbereiche und der Informatikabteilung sowie auf Kosten für externe Berater und externe Schulung. Die Kosten für die interne Schulung machen zirka 20%, Hardware- und Softwarekosten ebenfalls zirka 20% der bisher entstandenen Projektkosten aus.

Insgesamt sind inzwischen etwa 250 Mitarbeiter aus allen Firmenbereichen bei ihrer Arbeit vom neuen System direkt betroffen. Das System läuft auf einem Grossrechner, an dem insgesamt zirka 240 Terminals angeschlossen sind.

3 DIE SITUATION NACH DER SYSTEMEINFÜHRUNG

Im Rahmen der Projektrealisierung stand – wie bereits erwähnt – die Einführung des technischen Systems unter weitgehender Beibehaltung der or-

ganisatorischen Strukturen im Vordergrund. Die Form der Arbeitsteilung sowie die betrieblichen Abläufe wurden also mit dem neuen System kaum verändert, sondern lediglich stärker formalisiert und – vereinzelt – stärker automatisiert. Dies zeigt sich u.a. an der veränderten Arbeitssituation der Disponenten; genauere Lagerdisponierungen, On-line-Verfügbarkeitsprüfungen sowie präzisere Terminierungen sind nun zwar möglich, dies bedeutet jedoch auch, dass die Auftragsauslösung formalisierter und zeitaufwendiger wurde.

Die Ergebnisse einer schriftlichen Befragung unter Mitarbeitern aus produktionsvorgelagerten Bereichen zeigen, dass die Arbeitssituation nach der Systemeinführung als höher beanspruchend erlebt wird. So haben die Schwierigkeit der Arbeit, das Arbeitstempo, der Leistungsdruck, die nötige Konzentration und Genauigkeit sowie Belastung und Ermüdung aus der Sicht dieser Beschäftigten zugenommen.

Am PPS-System werden vor allem zu lange Antwortzeiten kritisiert. Die aufgabenbezogene Unterstützung des Systems sowie die Nützlichkeit der Hilfsinformationen werden mehrheitlich positiv bewertet, die Einschätzung der Systemflexibilität sowie -konsistenz ist sehr unterschiedlich. Die Ausbildung für die neue Arbeitssituation wird von den befragten Systembenutzern mehrheitlich als genügend bewertet.

Kommentare der befragten Mitarbeiter zur Veränderung haben mehrheitlich einen negativen 'Unterton'. Dies zeigt sich z.B. an folgenden Äusserungen: Seit der Einführung des PPS-Systems in unserer Firma "... läuft vieles etwas zähflüssiger als zuvor ...", "... hat sich die Ablauforganisation eher negativ verändert ... ", "... sind die Spannungen im zwischenmenschlichen Bereich gestiegen...", "... funktioniert ohne Computer nichts mehr", "… ist meine Motivation gesunken".

Im Rahmen von Interviews mit einzelnen Mitarbeitern wird sehr deutlich, dass neben positiven Erlebnissen mit und nach dem Projekt – z.B. das kooperative Verhalten in der neuen Arbeitssituation – zum Teil auch Unsicherheit und Angst bei den Mitarbeitern bestanden. Exemplarisch zeigt sich dies an Äusserungen wie: "Ich war drei Monate ohne Schlaf", oder: "Dies will ich nicht mehr erleben."

In bezug auf die betriebswirtschaftliche Situation bzw. die Zielerreichung nach der Systemeinführung hatten zunächst fehlerhafte Systemeinstellungen sowie Fehler im System selbst zur Folge, dass keine Aufträge aus dem System generiert werden konnten. In dieser als äussert hektisch und spannungsgeladen erlebten Zeit brach "die Auslastung der Werkstatt ziemlich zusammen". Nachdem diese Situation bereinigt war, führten ein 'denklogi-

scher Fehler' sowie unrealistische Planzahlen im System dazu, dass erheblich zuviel disponiert und gefertigt wurde.

Klagen des Lagerpersonals bei den Meistern über 'ein platzendes Lager' wurden an die Disponenten weitergetragen und von dort aus dem Verkauf mitgeteilt. Die Kommunikationsprozesse funktionierten in dieser Situation offensichtlich nicht wie vorgesehen; vom Verkauf wurden die hohen Lagerbestände fälschlicherweise mit saisonalen Schwankungen begründet.

Diese Fehldisposition hatte für die Kapazitätsplanung enorme Konsequenzen, da die 'fehlenden' Kapazitäten zirka sechs Monate nach der Einführung mit Überstunden sowie der Einstellung von Teilzeitmitarbeitern kompensiert wurden. Nachdem dieser Fehler schliesslich erkannt worden war, fehlte es an Aufträgen und die bestehende Überkapazität wurde durch Entlassung von Teilzeitmitarbeitern sowie Arbeitszeitreduktionen wieder abgebaut.

Ca. sechs Monate später zeigte sich erneut, dass beim Aufbau eines neuen Sortiments der Kapazitätsbedarf nicht eingeplant worden war und abgebaute Kapazitäten nun eigentlich wieder notwendig wären.

Bezogen auf die betriebswirtschaftlichen Projektziele musste ca. 12 Monate nach der Systemeinführung folgendes Fazit gezogen werden:

(1) Die Lagerbestände waren – wie dargestellt – erheblich gestiegen;

(2) die Termintreue hatte sich auf Kosten gestiegener Lagerbestände verbessert;

(3) die Durchlaufzeiten blieben grösstenteils unverändert.

Im Laufe des Jahres hatte sich zudem gezeigt, dass die System-'Performanz' nicht mehr ausreichte und die Antwortzeiten erheblich zu lang waren. System-'Tuning' zur Verbesserung der Performanz führte zu mehreren Systemabstürzen mit der Konsequenz, dass die Auftragsabwicklung in der Firma fast völlig zum Erliegen kam bzw. 'nichts mehr ging'.

4 PHASE III: DIE 'EHER ARBEITSORIENTIERTE KORREKTUR'

Einige systembezogene Änderungsmassnahmen sind zirka sechs Monate später erfolgt. Das System läuft nun mit besserer Performanz und Stabilität. Inzwischen wurde auch eine Analyse der bestehenden Abläufe und der technisch-organisatorischen Bedingungen in der Firma durchgeführt. Die Ergebnisse dieser Analyse brachten erwartungsgemäss eine Reihe von Problemen des betrieblichen Alltages zum Vorschein. So wurde deutlich, dass

in der Firma z.T. sehr ineffiziente Arbeitsstrukturen und Abläufe (Parallelität, Redundanz, zu viele Schnittstellen) praktiziert werden, die durch das neue PPS-System noch zementiert wurden. Einseitiges Abteilungsdenken bzw. mangelndes Verständnis einer gemeinsamen Aufgabe, schlechte Datenpflege – und somit schlechte Datenqualität – sowie unzureichende Nutzung der Systemfunktionalität sind einige Folgen dieser Situation.

Die dargestellten betriebswirtschaftlichen Ergebnisse nach der Systemeinführung wie auch die Ergebnisse dieser Analyse hatten in der Projektgruppe wie auch in einer inzwischen neu zusammengesetzten Geschäftsleitung die Einsicht verstärkt, dass die ursprünglich angestrebten betriebswirtschaftlichen Ziele mit dieser technisch-organisatorischen Lösung allein nicht erreicht werden können; das heisst, dass der Einsatz eines technisch zwar leistungsfähigen, integrierten PPS-Systems *ohne* Veränderung der bestehenden Organisationsstrukturen die Durchlaufzeiten nicht reduziert und die Termintreue nicht oder allenfalls auf Kosten hoher Lagerbestände erhöht. Im Zusammenhang mit diesen Überlegungen wurden u.a. die ursprünglich formulierten organisatorischen Ideen wieder aufgegriffen und spezifiziert. Einige der angestrebten Veränderungen sind bereits realisiert; andere befinden sich in der Realisierungsphase. Diese Veränderungsmassnahmen weisen in eine arbeitsorientierte Richtung und sind u.a. durch Merkmale wie Teile- und Produktsegmentierung, funktionale Integration sowie erweiterte Aufgabenspektren für die Beschäftigten gekennzeichnet.

5 FAZIT

Das beschriebene Fallbeispiel zeigt die Bedeutsamkeit arbeitsorganisatorischer Aspekte bei der Einführung komplexer Softwaresysteme auf. Im Rahmen des beschriebenen Projektes wurde u.a. versäumt, ein auf den Ergebnissen einer Ist- und Schwachstellen-Analyse basierendes Produktionskonzept unter Beteiligung der betroffenen Fachbereiche, der Informatikabteilung und der Geschäftsleitung zu entwickeln. Die fehlende Einigkeit in der Geschäftsleitung sowie unter den Abteilungsleitern über die geplante technisch-organisatorische Erneuerung war sicherlich einer der zentralen Gründe dafür, warum die Realisierung organisatorischer Innovationen mit den beschriebenen Problemen erst nach der Einführung des technischen Systems erfolgte.

Bei der Gestaltung der Projektorganisation wäre zudem eine umfassendere und repräsentativere Fachbereichsvertretung notwendig gewesen. Einigen der dargestellten Probleme hätte u.a. durch eine Integration von Vertretern anderer Bereiche wie z.B. aus dem Verkauf, der Beschaffung und der Fertigung in die Projektgruppe entgegengewirkt werden können. Das Durchführen von Informationsveranstaltungen, Workshops mit einer möglichst

grossen Anzahl von Projektbetroffenen zum Inhalt und Stand des Projektes schliessen sich als weitere Forderungen daran an. Vor einem soziotechnischen Hintergrund wäre für das beschriebene Vorhaben folgender, grob beschriebener, arbeitsorientierter Ablauf begründet gewesen:

(1) Ist- und Schwachstellen-Analyse der soziotechnischen Systeme,

(2) Entwicklung eines arbeitsorganisatorischen Soll-Konzeptes; Ableitung von Anforderungen bez. Technik und Mitarbeiterqualifikation (Produktionskonzept),

(3) Realisierung des Organisationskonzeptes – z.B. im Rahmen eines Pilotes in einer Abteilung – in 'konventioneller' Technikumgebung; Durchführung der notwendigen Qualifizierungsmassnahmen,

(4) Evaluation und Optimierung der neuen Arbeitsstrukturen,

(5) Entwicklung eines Feinkonzeptes für die Hard- und die Software,

(6) Auswahl, Anpassung und Implementierung der Hard- und der Software.

Ob ein solcher arbeitsorientierter Projektablauf dabei auch zur gleichen technischen Lösung geführt hätte, bleibt in diesem Zusammenhang eine offene Frage. Neben den dargestellten Versäumnissen und Problemen war das Projekt jedoch durch eine positive 'Wendemarke' gekennzeichnet: die Phase III mit der 'eher arbeitsorientierten Korrektur'. Die Erkenntnisse, die zu diesem 'Richtungswechsel' führten, wie auch die damit verbundenen Massnahmen eröffneten die Möglichkeit, dass technologische Innovationen auch mit organisatorischen Innovationen verbunden werden und die mit dem Projekt verbundenen betriebswirtschaftlichen Ziele erreicht werden können. In der Firma besteht nun die Einsicht, dass eine Vielzahl der bestehenden Probleme nur mit neuen Formen der Arbeits- und Organisationsgestaltung gelöst werden können. Dieser 'Richtungswechsel' wurde jedoch u.a. erst durch die Neubesetzung von Leitungsfunktionen ermöglicht.

ANHANG

Checkliste 1:

PRINZIPIEN UND HINWEISE FÜR VERÄNDERUNGSPROZESSE UND DIE BETEILIGUNG VON BENUTZERN BEI DER SOFTWAREENTWICKLUNG

Voraussetzung

Für einen erfolgreichen und effizienten Entwicklungsprozess müssen die (infra-)strukturellen Bedingungen in der EDV-, Entwicklungs-Abteilung stimmen; stark arbeitsteilige (Projektleiter, Analytiker, Programmierer, Codierer), funktionsbezogene (Analyse, Programmierung, Test, Wartung, Methodik) Strukturen, eine – bezüglich Werkzeugen und deren Integration – ungenügende Entwicklungsumgebung und bürokratische Vorschriften für die Projektabwicklung ('Formularkrieg' bei Anträgen, Genehmigungen, Berichterstattung usw.) verhindern ein kreatives, innovatives Vorgehen und erschweren die Beteiligung von Benutzern. *Auch die Entwickler müssen gute Arbeitsbedingungen haben, damit sie im Projekt Engagement zeigen und effizient arbeiten können!*

Prinzip 1

Ausgehend vom Soll-Konzept der Arbeitsorganisation sind die technischen Unterstützungsbedürfnisse abzuleiten, die Funktionsteilung zwischen Mensch und Computer (und damit die Systemfunktionalität!) zu konzipieren und anschliessend die Benutzungsoberfläche zu entwerfen.

Prinzip 2

Statt länger dauernde Grossprojekte besser kürzere Teilprojekte mit bald sichtbarem Erfolg für die Beteiligten realisieren! Grossprojekte in überschaubare Einheiten aufteilen! Die Motivation der Beteiligten nimmt mit der Zeit ab, wenn kein Resultat der Arbeit ersichtlich wird.

Prinzip 3

Zur Vermeidung oder Überwindung von Ungewissheiten, Ängsten und Widerständen sollten die von der Veränderung betroffenen Mitarbeiter frühzeitig und fortlaufend informiert, in den Veränderungsprozess einbezogen und gründlich ausgebildet werden: die Betroffenen zu Beteiligten machen!

Das Fachwissen der zukünftigen Benutzer durch deren frühzeitigen Einbezug in die Softwareentwicklung nutzen!
Der Einbezug der Benutzer darf aber nicht auf eine Pseudopartizipation hinauslaufen, denn sie wird meistens durchschaut und führt zu keinen längerfristig befriedigenden Ergebnissen.

Prinzip 4

Bei der Organisation der Benutzerbeteiligung ist unbedingt auf eine *repräsentative Vertretung* verschiedener Fachabteilungen, Hierarchieebenen und Benutzergruppen zu achten!

Prinzip 5

Die am Projekt mitwirkenden *Benutzer* müssen zu einem Teil ihrer Arbeitszeit von ihren täglichen Aufgaben *freigestellt* werden, um Überlastungen zu vermeiden und ihnen eine engagierte Mitarbeit zu ermöglichen. Projektaufgaben dürfen ihnen nicht als zusätzliche Arbeit aufgebürdet werden, da dies in der Konsequenz allmählich dazu führt, dass die Projektarbeit als lästige Pflichtübung betrachtet und vernachlässigt wird!

Prinzip 6

Die *Erwartungen* aller Beteiligten in bezug auf die Ziele des Projektes müssen auf einem *realistischen Niveau* gehalten werden; es dürfen keine überzogenen Versprechungen gegeben werden, die dann unter Umständen nicht eingehalten werden können! Enttäuschte Erwartungen führen zu Frustration, die sich in Passivität und Widerstand äussern und die Zusammenarbeit gefährden kann.

Prinzip 7

Es kann gefährlich sein, Projektmanagement-'Rezepte' ungeprüft zu übernehmen; besser ist es, eine dem Betrieb, dem Vorhaben sowie den personellen und infrastrukturellen Voraussetzungen *angepasste Projektorganisation* zu entwickeln!

Prinzip 8

Benutzer und Entwickler sind durch eine *komplementäre Ausbildung* auf die Zusammenarbeit vorzubereiten; die Benutzer müssen über Möglichkeiten und Grenzen des technischen Systems orientiert werden, die Entwickler

müssen die Aufgaben und Arbeitsabläufe der Benutzer gut kennen! Dies erleichtert auch das gegenseitige Verständnis und die Kommunikation!

Prinzip 9

Für *Analyse und Anforderungsermittlung* ist *mehr Zeit* als üblich einzukalkulieren! Auf *neue Wünsche und Änderungsvorschläge* der Benutzer im Laufe des Projektes muss man *gefasst sein* sowie offen und flexibel darauf reagieren!
Benutzerbeiträge nicht a priori mit technischen Argumenten abweisen ("sowieso nicht machbar"), sondern ernsthaft prüfen und allenfalls Realisierungsprobleme erläutern, denn: mehrfach abgewiesene Vorschläge der Benutzer führen zu Passivität und Resignation ("... es hat ja doch keinen Sinn...")! Benutzerbeteiligung darf keine Alibiübung sein.

Prinzip 10

Anstelle schwer verständlicher Pflichtenhefte, Systembeschreibungen usw. *Prototypen erstellen*, den Benutzern erläutern, ihnen Zeit für eine eingehendere Beschäftigung mit dem Prototyp geben und dann eine Besprechung oder einen Workshop durchführen! 'Ein Bild sagt mehr als tausend Worte'!

Prinzip 11

Nicht ausgetragene, durch Gespräche bewältigte *Konflikte* in Projekt- oder Arbeitsgruppen können das gesamte Projekt gefährden. Ebenso können andauernde Kommunikationsprobleme zwischen Benutzern und Entwicklern eine effiziente, fruchtbare und persönlich befriedigende Zusammenarbeit verunmöglichen. In diesen Fällen kann ein begleitendes *Training sozialer Fertigkeiten und der Konfliktbewältigung* hilfreich sein!

Projektorganisation

(1) Sind *alle relevanten Interessengruppen* in der Projektorganisation vertreten?
[ ] Management
[ ] Spezialisten (z.B. einer Stabsabteilung)
[ ] betroffene Fachabteilungen
[ ] externe Berater
[ ] Betriebsrat, Personalkommission
[ ] weitere: .....

(2) Ist die Organisation der *Benutzerbeteiligung* (bezüglich Form, Grad, Inhalt, Methoden, Zeitpunkt) *sichergestellt*?
(3) Sind Funktion, Aufgaben, Kompetenzen und Verantwortlichkeiten der gebildeten Gremien und deren Zusammenwirken – vor allem bezüglich Entscheidungsprozessen – definiert? Gibt es Unklarheiten bezüglich Zuständigkeiten, gibt es Doppelspurigkeiten oder Lücken?
(4) Sind Projekt- und Arbeitsgruppen ihren Aufgaben entsprechend personell richtig besetzt?
(5) Sind Inhalt und Zielsetzungen des Projektes definiert? Sind sie allen Beteiligten bekannt und besteht ein Konsens darüber?
(6) Besteht ein *Konzept über die Projektabwicklung* mit Zeitplan, Ressourcen, Aktivitäten usw.?
(7) Sind genügend *Zeitreserven* eingeplant? Ist speziell für Analyse, Anforderungsermittlung und Grobkonzept ausreichend Zeit vorgesehen, so dass auch mehrere Lösungsvarianten entwickelt werden können?
(8) Ist ein *Informationsverteilungsnetz* vorhanden? Sind Kommunikationskanäle geschaffen und auch allen Beteiligten bekannt?
(9) Ist ein *Dokumentationssystem* eingerichtet?
(10) Sind alle Voraussetzungen methodischer und technischer Art für den Projektbeginn überprüft und erfüllt?
(11) Sind alle Beteiligten auf dem *erforderlichen Ausbildungsstand*?

Information

(1) Durch frühzeitig beginnende, fortlaufende Information wird der Entstehung von Ungewissheiten und Gerüchten vorgebeugt und Transparenz geschaffen.
(2) Zur *Verbreitung der Information und zur Ermöglichung von Rückmeldungen* sind Kommunikationskanäle und ein Informationsverteilungsnetz zu schaffen. Möglichkeiten:
    • periodische Information in der 'Hauszeitung',
    • periodische, projektspezifische Mitteilungen, z.B. als Bulletins,
    • Mailbox, Anschlagbrett,
    • persönliches Anschreiben, Gespräch,
    • Abteilungs-, Betriebsversammlung,
    • Vertiefung der Information in Seminaren,
    • Diashows, Filme und Modelle zur Veranschaulichung.
(3) Nicht nur die vom Projekt direkt Betroffenen, sondern auch die indirekt Betroffenen sollen über die geplanten Neuerungen angemessen informiert werden.

(4) Das *Schwergewicht* der Information soll nicht auf der detaillierten Darstellung der technischen Veränderung liegen, sondern vor allem deren Bedeutung für die *Veränderungen der Arbeitsorganisation und der Arbeitsrolle der Mitarbeiter* darlegen.
(5) Die *Erstinformation* soll den Mitarbeitern global Auskunft geben über
- die Gründe, die zum Projekt geführt haben,
- die möglichen Auswirkungen des Projektes in personeller, arbeitsorganisatorischer und technischer Hinsicht,
- das weitere Vorgehen und Möglichkeiten der Mitwirkung.

(6) Die *fortlaufende Information* soll die Mitarbeiter orientieren über
- den Stand des Veränderungsprozesses,
- entwickelte Konzepte,
- getroffene Entscheide und daraus abgeleitete, bevorstehende Aktivitäten.

Partizipation

(1) Der Einbezug der Benutzer sollte schon in der Vorbereitungs- und Planungsphase erfolgen, damit ihre Bedürfnisse berücksichtigt und ihr Fachwissen genutzt werden kann. Dies reduziert auch die Wahrscheinlichkeit, dass nachträgliche Korrekturen notwendig werden.
(2) Wer sind die zukünftigen Benutzer, was sind ihre Aufgaben? Sind die Benutzervertreter repräsentativ für die gesamte zukünftige Benutzerpopulation?
(3) Werden die Benutzervertreter für die Projektarbeit freigestellt, das heisst, von ihren sonstigen Aufgaben zum Teil entlastet?
(4) Zur *Beteiligung* der Benutzer gibt es *verschiedene Möglichkeiten*:
- mündliche und schriftliche Umfragen,
- die Mitarbeit in Projektgruppen,
- die vollständige Übertragung von Teilaufgaben an Mitarbeiter in Arbeitsgruppen,
- den Einsatz von Kontaktpersonen,
- periodisch durchgeführte Workshops usw.

Ausbildung

(1) Eine *frühzeitige* Ausbildung von *Benutzern und Entwicklern* ist die Grundlage für eine aktive, kompetente Mitwirkung und effiziente Arbeit im Projekt. Inhalte können sein:
- Organisation,
- soziale Fertigkeiten,
- Methoden, Fachwissen,

- EDV, Technik,
- Ergonomie,
- Präsentationstechniken.

(2) Eine gründliche Schulung der operationellen Systembenutzung vermindert die Befürchtungen der Mitarbeiter, die zukünftigen Anforderungen mit den bisherigen Kenntnissen nicht mehr bewältigen zu können.
Zusätzlich sollte den Benutzern ein Grundlagenwissen vermittelt werden, das ihnen zu einem besseren Verständnis der prinzipiellen Funktionsweise des technischen Systems verhilft. Die Benutzer werden sich dadurch der Technik gegenüber weniger hilflos und ausgeliefert fühlen.

(3) Die Schulung sollte den *Vorkenntnissen der Teilnehmer angepasst* sein und auf die *konkreten Aufgaben und Arbeitsabläufe bezug nehmen.*

(4) Die Ausbildungsinhalte sollten möglichst *direkt*, das heisst ohne längere zeitliche Verzögerung, in die Praxis *umsetzbar* sein.

(5) Die Einrichtung von Übungsarbeitsplätzen vor der Systemeinführung ermöglicht eine spielerische (und damit angstfreie) Angewöhnung an das zukünftige technische System.

(6) Ein verständlich geschriebenes, gut aufgebautes, nicht zu umfangreiches *Benutzerhandbuch mit Beispielen*, Erklärungen von Besonderheiten und Hinweisen auf spezielle Arbeitshilfen kann die Systembenutzung wesentlich erleichtern.
Zusätzlich empfehlenswert ist die Abgabe eines Faltblattes (im Sinne eines 'Quick Reference Guides') mit Kurzbeschreibungen der wichtigsten Befehle, der Funktionstastenbelegung, 'short cuts', häufig gebrauchten Code keys usw.

(7) Die Ausbildung sollte auch eine *Aufklärung über Möglichkeiten der individuellen Anpassung* des Computerarbeitsplatzes (Verstellbarkeit von Bildschirm, Stuhl, Tisch usw.), des ermüdungsarmen Arbeitens (Vermeidung von Blendung, Zwangshaltungen usw.), Einstellung von Farben, Kontrast, Helligkeit usw. enthalten.

Einführung

(1) Gerade bei grösseren Projekten sollte die Einführung des Systems schrittweise (sinnvoll abgegrenzte (Teil-)Anwendungen, Module) erfolgen. Die Veränderung wird dadurch für die Mitarbeiter leichter überblickbar.

(2) Jeder Teilschritt sollte für die Mitarbeiter mit *Erfolgserlebnissen* verbunden sein und Vorteile erkennbar werden lassen.

(3) Die Veränderung verlangt von allen Beteiligten einen stetigen Lernprozess; ist deshalb eine Evaluation des Systems nach der Einführung –

das heisst, nachdem es einige Zeit im produktiven Einsatz stand – vorbereitet, um eventuell noch vorhandene Mängel erkennen (und korrigieren!) zu können?

(4) Während und auch einige Zeit nach der Einführung müssen die Benutzer die Möglichkeit haben, bei Benutzungsproblemen *menschliche Unterstützung* (z.B. über eine Hotline) zu erhalten; ist dafür genügend personelle Kapazität vorgesehen und ist den Benutzern die Anlaufstelle auch bekannt? On-line-Tutorials, Hilfefunktionen und Handbücher bieten dafür *keinen Ersatz!*

Prototyping

(1) Sind *Werkzeuge* ('tools') für die Erstellung von Prototypen *vorhanden?*

(2) Herrscht bei allen Beteiligten *Klarheit über die Fragen,* die mit dem Prototyp beantwortet werden sollen?

(3) Gibt es ein *Konzept für die Austestung* von Prototypen durch Benutzer?

(4) Sind *realistische Aufgaben* (mit Testdaten) formuliert?

(5) Wird der Test durch Mitarbeiter durchgeführt, die *repräsentativ* sind für die zukünftigen Benutzer?

(6) Ist eine *schnelle Änderung von Prototypen* gewährleistet? (Eine allzu frühe Fixierung auf eine Lösung verhindert die Entwicklung alternativer, eventuell besserer Varianten!)

Checkliste 2:

VEREINBARUNGEN FÜR EINEN PARTNERORIENTIERTEN DEMOKRATISCHEN GRUPPENPROZESS

Für die erfolgreiche Arbeit in einer Gruppe können die folgenden Aspekte als wichtigste Voraussetzungen gelten:

	Vereinbarung auf		Vermeidung von
+	Persönliches Engagement jedes Teilnehmers: d.h., Einigkeit über die Zielsetzung(en).	–	Vorgefassten Meinungen, funktionellen Fixationen: d.h., kein Festhalten an persönlichen einseitigen Erfahrungen.
+	Jeder soll reden können, jeder soll aktiv sein: d.h., niemandem die Meinung aufdrängen (–> ausdiskutieren); nicht vorzeitig nachgeben, nur um Einigkeit zu erreichen (kein 'Kuhhandel'); 'Störungen' anmelden.	–	Ungünstigen äusseren Bedingungen: d.h., kein Zeitdruck; keine Störungen von aussen; Gruppe nicht zu gross; alle Teilnehmer sind anwesend.
+	Jeder ist Experte auf seinem Gebiet, jeder ist gleichberechtigt: d.h., abweichende Meinungen sind nicht störend, sondern ein konstruktiver Beitrag.	–	Autoritätsdenken, Festhalten an eingefahrenen Rollen: d.h., der Leiter drängt sich nicht in den Vordergrund; der Moderator achtet auf Demokratie.
+	Alle Hinweise aufgreifen, nichts untergehen lassen: d.h., Trennung von 'Probleme sammeln, Ideen finden' und 'Probleme diskutieren, Ideen bewerten'; alles für jeden sichtbar registrieren.	–	Abschottung gegenüber neuen Ideen: d.h., keine Killerphrasen (verbal und nonverbal), wie z.B. "... ist technisch nicht machbar", "... das haben wir noch nie so gemacht".
+	Das Ergebnis der Gruppenarbeit ist Eigentum der Gruppe: d.h., kein individuelles Urheberrecht.		

Übersicht 1:

ÜBERSICHT ÜBER INDIKATOREN FÜR DAS SCHEITERN VON SOFTWAREPROJEKTEN

Nach Bartsch-Spörl (1991, S. 304ff) gibt es die folgenden typischen Situationen, die als Indikatoren für 'Projektkrankheiten' gelten können:

1. *Alle sind sich einig, dass das Projekt gemacht werden sollte, aber keiner will die Verantwortung dafür übernehmen.*
   Eine solche Situation deutet darauf hin, dass eventuell a) das Risiko des Scheiterns insgesamt noch zu hoch ist und/oder dass b) die Übernahme der Verantwortung nicht attraktiv ist, weil sie z.B. ein Übermass an Belastung mit Pflichten ohne ausreichende Rechte bedeuten würde. Im Fall A ist die Zeit für das Projekt noch nicht reif; im Fall B müssen die Rahmenbedingungen geändert werden.
2. *Die Geschäftsleitung hat ein Projekt 'angeordnet', das die Anwender nicht wollen.*
   Hier helfen nur Gespräche mit beiden Seiten mit dem Ziel, einen für alle Beteiligten gangbaren Weg zu einem eventuell veränderten Projektziel zu finden. Wenn dies nicht möglich ist, sollte man von der Durchführung dieses Projekts Abstand nehmen.
3. *Ein Projekt ist von den Ressourcen her zu knapp budgetiert worden.*
   Hier hilft entweder a) eine entsprechende Aufstockung der Ressourcen oder b) der Verzicht auf einen erheblichen Teil der Funktionalität. Ist Fall A nicht möglich und Fall B nicht sinnvoll, dann sollte man das Projekt verschieben.
4. *Die EDV-Leute verfolgen ein rein technikorientiertes Ziel (z.B. die Evaluation von CASE-Werkzeugen) und wollen das mit dem Budget der Anwender bezahlen.*
   Wenn das Projekt z.B. eine für die Firma insgesamt wichtige Infrastrukturmassnahme ist, dann muss sich das in einer veränderten Art der Finanzierung niederschlagen; wenn es jedoch niemandem einen ersichtlichen Nutzen bringt, dann gehört es beendet.
5. *Für die geplante Anwendung ist eine ungeeignete Entwicklungsphilosophie gewählt worden (z.B. Entwicklung eines Management-Informationssystems in COBOL ohne Datenbank und ohne flexible Abfragesprache).*
   Wenn der Anwendernutzen mit der gewählten Entwicklungsphilosophie nicht erreicht werden kann, muss die technologische Basis gewechselt

werden – dies ist zwar manchmal teuer, aber jedenfalls besser als eine unbrauchbare Anwendung.

6. *Der Zeitplan des Projektes gerät ins Wanken (z.B. aufgrund von nicht kalkulierten Personalfehlzeiten).*
   In solchen Fällen sollte man sich keine Illusionen machen, dass man die Rückstände leicht wieder einholen kann; man sollte lieber gleich mit den Anwendern über einen späteren Einführungstermin oder über eine erste Version mit reduzierter Funktionalität zu sprechen beginnen.

7. *Es gibt keinen einzigen 'Anwendungsexperten' in der Projektgruppe (z.B. weil die Fachabteilung unter akutem Personalmangel leidet).*
   Diese höchst bedenkliche Situation ist als untragbar, in wichtigen Projekten sogar als unverantwortlich anzusehen; hier gilt es unbedingt eine Lösung zu finden, die die Fachabteilung aktiv in das Projekt mit einbezieht.

8. *Die im Projekt verwendete Methodik stösst auf Ablehnung bei den Projektmitarbeitern.*
   Dieser äusserst alarmierende Indikator kann bedeuten: Fall A – die Qualifikation für den Einsatz der Methodik fehlt; Fall B – die 'verordnete' Methodik gehört einer anderen 'Generation' an und setzt andere als von der zu entwickelnden Anwendung benötigte Prioritäten; Fall C – es wird ein Machtkampf zwischen altgedienten Softwareentwicklern und einem neuen Vorgesetzten ausgetragen. Die Lösung heisst: im Fall A die Projektmitarbeiter qualifizieren, im Fall B die Methodik wechseln und im Fall C eventuell abwarten, bis sich die Wogen geglättet haben.

9. *Der Entwurf hat sich 'festgefahren', das heisst, berechtigte Anforderungen lassen sich nicht mehr berücksichtigen.*
   Dies ist eine Sackgasse, aus der das Projekt unbedingt herausgeholt werden muss, ggf. um den Preis eines zusätzlichen Iterationsschrittes für den gesamten Entwurf. Es ist in diesem Fall zunächst besonders wichtig, die Ursachen für die fehlende Flexibilität zu finden, denn dann kann man sinnvoll entscheiden, was beim nächsten Anlauf anders gemacht werden muss.

10. *Die Qualitätssicherung (QS) wird von der Projektleitung auf die leichte Schulter genommen.*
    Die Folgen dieses unverantwortlichen Verhaltens werden die Organisation und die späteren Anwender 'schmerzhaft' zu spüren bekommen. An der notwendigen Qualität zu sparen ist der denkbar teuerste Fehler.

Übersicht 2:

ÜBERSICHT ÜBER HILFSMITTEL ZUR ENTWURFSUNTERSTÜTZUNG

'Dynamic Rules for User Interface Design' (DRUID) als Hyper-Card-Stapel für den Apple-Macintosh aus dem Jahre 1989 umfasst alle Gestaltungshinweise und Entwurfsrichtlinien der Sammlung von Smith und Mosier (1986). Diese hypertextbasierte Software ist ebenso wie die Papierversion der 'Guidelines for Designing User Interface Software' von Smith und Mosier zu beziehen bei:

U.S. Department of Commerce, National Technical Information Service (NTIS), Springfield, VA 22161, USA.

Die entsprechende IBM-kompatible PC-Version als Hypertextsystem ist unter dem Namen 'Navi Text SAM' bei der folgenden Adresse beziehbar:

North Lights – Software Corporation, 24A Pilgrim Drive, P.O. Box 1056, Westford, MA 01886, USA.

Für die Unterstützung der Analyse von Benutzerverhalten, welches mit einem Videorekorder aufgenommen wurde, steht das folgende Softwarewerkzeug zur Verfügung: 'DRUM – The Diagnostic Recoder for Usability Measurement', beziehbar bei:

National Physical Laboratory, DITC HCI Group, Teddington, Middlesex, TW11 0LW, United Kingdom, Telefon: +44 81 943 6097.

Als geeignete Hilfsmittel zur schnellen Entwicklung von Masken und Oberflächen können die folgenden Softwarewerkzeuge angegeben werden:

Animation Works (1991) Gold Disk Inc., P.O. Box 789, Streetsville, Mississauga, Ontario, Canada L5M 2C2.

Astound (1993) Gold Disk Inc., P.O. Box 789, Streetsville, Mississauga, Ontario, Canada L5M 2C2.

Hypercard (1992) Claris International Inc., 250 The Esplanade, Suite 202, Toronto, Ontario M5A 1JS. Telefon (416) 941 9611.

IconAuthor (1989) ADI GmbH, Hardeckstr. 5, D-75021 Karlsruhe. Telefon (721) 57 0000.

IconAuthor (1991) AimTech Europe Limited, 1 Carthusian Street, London EC1M 6EB (UK). Telefon (071) 250 1225.

MacroMind Director (1990) MacroMind, 410 Townsend, Suite 408, San Francisco, CA 94107 (USA). Telefon (415) 442 0200.

NeXT Step (1991) NeXT Inc., 900 Chesapeake Drive, Redwood City, CA 94063 (USA).

NeXT Interface Builder (1991) NeXT Inc., 900 Chesapeake Drive, Redwood City, CA 94063 (USA).

Prototyper (1990) Smethers & Barnes Inc., P.O. Box 639, Portland, Oregon 97207 (USA).

ToolBook (1989) ADI GmbH, Hardeckstr. 5, D-75021 Karlsruhe. Telefon (721) 57 0000.

ToolBook (1989) Asymetrix Corp., 110 Avenue N.E., Suite 717, Bellevue, Washington 98004 (USA).

VisualBasic (1993) Microsoft Corp., One Microsoft Way, Redmond, WA 98052-6399 (USA).

UIMX (1990) Visual Edge Software Ltd., 3870 Cote Vertu, Montreal, Quebec, Canada H4R 1V4, Telefon (514) 332 6430.

WindowsMAKER (1991) Blue Sky Software Corp., 2375 East Tropicana Avenue, Suite 320, Las Vegas, NV 89119 (USA). Telefon (702) 465 6365.

Folgende Guidelines, Leitfäden, Checklisten usw. sind im Buchhandel erhältlich:

Apple® (1990) Human Interface Guidelines: The Apple Desktop Interface. Amsterdam: Addison Wesley.

IBM® Systems Application Architecture (1991) Common User Access – Guide to User Interface Design. (Bestell-Nr.: SC34-4289-00), IBM Corporation, Department T45, P.O. Box 600 000, Cary, North Carolina 27512-9968, USA.

IBM® Systems Application Architecture (1989) Common User Access – Basic Interface Design Guide. (Bestell-Nr.: SC26-4583-0), IBM Corporation, Department T45, P.O. Box 600 000, Cary, North Carolina 27512-9968, USA.

IBM® Systems Application Architecture (1991) Common User Access – Advanced Interface Design Reference. (Bestell-Nr.: SC34-4290-00), IBM Corporation, Department T45, P.O. Box 600 000, Cary, North Carolina 27512-9968, USA.

INRA (1994) Der Interface Ratgeber: ein Informations- und Beratungssystem zur Software-Ergonomie. Zu beziehen bei: Prof. Dr. H. Wandke, Institut für Psychologie, Humboldt-Universität, Oranienburger Str. 18, D-10178 Berlin, Telefon: +49-30-2805115, Fax: +49-30-2829179.

ISO 9241/10 (1993) Fragebogen zur Beurteilung von Software auf der Grundlage der Internationalen Ergonomie-Norm ISO 9241/10. Dr. Prümper & Partner, Holzhofenstrasse 8, D-81667 München.

MUSiC (1993) Metrics for Usability Standards in Computing. Zu beziehen bei: National Physical Laboratory, DITC HCI Group, Teddington, Middlesex, TW11 0LW, United Kingdom, Telefon: +44-81-943 6097.

OSF/Motif™ (1990) User's Guide. Englewood Cliffs: Prentice Hall.

Siemens Nixdorf Informationssysteme AG (1992) Alpha-Styleguide – Richtlinien zur Gestaltung von Zeichenorientierten Benutzungsoberflächen. (Bestell-Nr.: U8556-J-Z147-1), Otto-Hahn-Ring 6, D-8000 München 83.

Siemens Nixdorf Informationssysteme AG (1992) Alpha-Styleguide-Checkliste V1.0. (Bestell-Nr.: U8557-J-Z147-1), Otto-Hahn-Ring 6, D-8000 München 83.

Siemens Nixdorf Informationssysteme AG (1992) Styleguide V1.0 – Richtlinien zur Gestaltung von Benutzungsoberflächen. (Bestell-Nr.: U6542-J-Z97-1), Otto-Hahn-Ring 6, D-8000 München 83.

Siemens Nixdorf Informationssysteme AG (1992) Styleguide-Checkliste. (Bestell-Nr.: U6615-J-Z97-1), Otto-Hahn-Ring 6, D-8000 München 83.

Smith S. & Mosier J. (1986) Guidelines for Designing User Interface Software. (MITRE Technical Report, Document No. AD-A177 198). Bedford MA (USA).

Software Checker (1992; version 2.0) An aid to the critical examination of the ergonomic properties of software. Zu beziehen bei: The Swedish Confederation of Professional Employees, P.O. Box 5252, S-102 45 Stockholm, Sweden, Telefon: +46-8-782 91 00.

SUMI (1993) Software Usability Measurement Inventory. Zu beziehen bei: Human Factors Research Group, Department of Applied Psychology, University College, Cork, Ireland, Telefon: +353-21-276 871.

Folgende Bücher zur softwareergonomischen Bildschirmgestaltung sind im Buchhandel erhältlich:

AWV Vordrucke 9 (1992) Arbeitshilfen für Anwenderfreundliche Bildschirm-Eingabemasken. Arbeitsgemeinschaft für wirtschaftliche Verwaltung e.V., Düsseldorfer Strasse 40, Postfach 5129, D-6236 Eschborn. (Bestell-Nr. 02-501), Telefon (06196) 495 388.

Baitsch C., Katz C., Spinas P. & Ulich E. (1989) Computerunterstützte Büroarbeit. Zürich: Verlag der Fachvereine.

Balzert H., Hoppe H. U., Oppermann R., Peschke H., Rohr G. & Streitz N. A. (1988; Hrsg.) Einführung in die Software-Ergonomie. (Reihe Mensch Computer Kommunikation – Grundwissen 1; Balzert, H., Hrsg.). Berlin: de Gruyter.

Brown C. M. (1989) Human-Computer Interface Design Guidelines. Norwood: Ablex.

Duell W. & Katz C. (1990) Ratgeber Bildschirmarbeit. (Schriftenreihe der Bundesanstalt für Arbeitsschutz, Forschungsanwendung FA 24). Bremerhaven: Verlag für neue Wissenschaften.

Galitz W. (1989) Handbook of Screen Format Design. P.O.Box 181, Wellesley, MA 02181 (USA).

Hoffmann T., Klose H-G. & Martin H. (1989) Handbuch zur softwareergonomischen Gestaltung von Bildschirmmasken. (VDI Fortschrittsberichte Reihe 10: Informatik/Kommunikationstechnik Nr. 103) Düsseldorf: VDI Verlag.

Lauter B. (1987) Software-Ergonomie in der Praxis. Wien: Oldenbourg.

Mayhew D. J. (1992) Principles and Guidelines in Software User Interface Design. Englewood Cliffs: Prentice Hall.

Rödiger K-H., Hampe-Neteler W. & Piepenburg U. (1991) Software-Ergonomie: Gestaltungsgrundsätze der DIN-Norm 66234 Teil 8 und ihre Umsetzung. Oberhausen: Technologie Beratungsstelle des DGB.

Spinas P., Troy N. & Ulich E. (1983) Leitfaden zur Einführung und Gestaltung von Arbeit mit Bildschirmsystemen. München: CW.

Wandmacher J. (1993) Software-Ergonomie. (Mensch-Computer-Kommunikation: Grundwissen, Band 2), Berlin New York: de Gruyter.

Ziegler J. & Ilg J. (1993; Hrsg.) Benutzergerechte Software-Gestaltung. München: Oldenbourg.

Zur Beurteilung und Einschätzung der verschiedenen Verfahren und Gestaltungsansätze sei auf die folgende Arbeit verwiesen:

Hampe-Neteler W. & Rödiger K-H. (1991) Software-Ergonomie: Verfahren der Evaluierung und Standards zur Entwicklung von Benutzungsoberflächen. (Fachbereich Mathematik/Informatik, Bericht Nr. 2/92). Bremen: Universität Bremen.

Übersicht 3:

ARBEITSANALYSEVERFAHREN

(in Anlehnung an Schüpbach 1993, S. 180–181)

Verfahren, Autor(en)	Bewertungskriterien, Hauptdimensionen	Anwendungsbereich
1. Beobachtungsinterviews: **a) Tätigkeitspsychologisch/handlungstheoretisch fundierte Verfahren**		
TBS-L – Tätigkeitsbewertungssystem (Langform) Hacker, Iwanova und Richter (1983)	Persönlichkeitsförderlichkeit: A: Organisatorische und technische Determinanten vollständiger Tätigkeiten B: Kooperation und Kommunikation C: Verantwortung D: Erforderliche psychische Prozesse und Repräsentationen E: Qualifikations- und Lernerfordernisse	Gestaltung industrieller Montage-, Bedien- und Überwachungstätigkeit: – Schaffung vollständiger Tätigkeitsinhalte – Schaffung von Handlungsspielräumen für die selbständige Planung und Ausführung komplexer Arbeitsabläufe
TBS-GA – Tätigkeitsbewertungssystem (geistige Arbeit) Rudolph, Schönfelder und Hacker (1987)	wie TBS-L	Wie TBS-L, aber für überwiegend geistige Arbeitstätigkeiten mit und ohne Rechnerunterstützung
SABA – Spezielle Analyse belastender Arbeitsfaktoren Richter, Heimke und Malessa (1988)	Beeinträchtigungslosigkeit: A: Unmittelbar gestaltbare Aufgabenmerkmale (Anzahl von Teiltätigkeiten, sequentielle Vollständigkeit, Bewegungsvielfalt, Zyklusdauer, Rückmeldung, Qualitäts-, Quantitätskonflikte B: Mittelbar gestaltbare Aufgabenmerkmale (soziale Konfliktaustragung, Kooperation, Entscheidungen, Planen, Niveau kognitiver Anforderungen, Qualifikationsanspruchnahme)	Gestaltung industrieller Montage-, Bedien- und Überwachungstätigkeit: – Sicherung vollständiger Tätigkeitsinhalte – Vermeidung von Regulationsbeeinträchtigungen
VERA – Verfahren zur Ermittlung von Regulationserfordernissen in der Arbeitstätigkeit Volpert et al. (1983)	Persönlichkeitsförderlichkeit: Niveau der Handlungsregulation nach einem Zehn-Stufen-Modell psychischer Regulationserfordernisse	Gestaltung industriell-gewerblicher Arbeitstätigkeit: Schaffung von Arbeitsaufgaben mit hohen kognitiven Regulationserfordernissen

KABA – Kontrastive Aufgabenanalyse im Büro Dunckel et al. (1993)	Humankriterien: Entscheidungsspielraum, Kommunikation, Belastungen, Zeitspielraum, Variabilität, Kontakt, Körperliche Aktivität, Strukturierbarkeit	Gestaltung von EDV-gestützten Arbeitssystemen vor dem Hintergrund der Humankriterien
RHIA – Regulationshindernisse in der Arbeitstätigkeit Leitner, Volpert, Greiner, Weber und Hennes (1987)	Beeinträchtigungslosigkeit: Regulationsbehinderungen als A: Regulationshindernisse (Erschwerungen, Unterbrechungen) B: Regulationsüberforderungen (aufgabenimmanent, aufgabenspezifisch)	Gestaltung industriell-gewerblicher Arbeitstätigkeit: Abbau psychisch belastender Arbeitsaufgaben- und -bedingungsmerkmale
ISTA – Instrument zur stressbezogenen Arbeitsanalyse (Beobachtungsversion) Semmer (1984)	Beeinträchtigungslosigkeit: A: Stressoren (unangemessener Regulationsaufwand, Regulationsunsicherheit), Zielunsicherheit B: Allgemeine Regulationsanforderungen und Ressourcen	Vergleich von Belastungsstrukturen bei unterschiedlichen gewerblichen Arbeitstätigkeiten
TAI – Tätigkeitsanalyseinventar Frieling, Kannheiser, Facaoaru, Wöcherl und Dürholt (1984)	Modularer Aufbau mit vier Teilverfahren zu den Problemschwerpunkten: I Emotional beanspruchungsrelevante Tätigkeitsbedingungen. II Kognitiv beanspruchungsrelevante Tätigkeitsbedingungen (informatorische Belstungen) III Qualifikationsanfoderungen IV Erfolgte und zu erwartende Veränderungen	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): Bildung und Vergleich von Bedingungs- und Anforderungskonfigurationen; Ableitung von Gestaltungs- und Qualifizierungserfordernissen; Evaluation von Veränderungsprozessen
b) Arbeitswissenschaftlich/verhaltenstheoretisch fundierte Verfahren		
FAA – Fragebogen zur Arbeitsanalyse Frieling und Hoyos (1978)	Angemessene Belastung/ Beanspruchung: - Informationsaufnahme und -verarbeitung - Arbeitsausführung (Arbeitsmittel, Bedienelemente, manuelle Tätigkeiten) - Umgebungseinflüsse und besondere Arbeitsbedingungen	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): vergleichende Arbeits- und Berufsanalysen; Klassifikation von Arbeitstätigkeiten und Berufen; Ableitung von Eignungsvoraussetzungen; Entwicklung von Lern- und Trainingsprogrammen
AET – Arbeitswissenschaftliches Erhebungsverfahren zur Tätigkeitsanalyse Rohmert und Landau (1979)	Angemessene Belastung/Beanspruchung: Formen menschlicher Arbeit in Arbeitssystemen (energetisch-effektorische, informatorische Tätigkeitselemente); Arbeitsobjekte und Arbeitsmittel; Umgebungseinflüsse (Beleuchtung, Lärm, Vibration, Klima, Arbeitssicherheit); organisatorische und wirtschaftliche Aspekte	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): – vergleichende Arbeitssystembeschreibungen – Ableitung von Anforderungen

2. Schriftliche Befragungen (Fragebogen)		
JDS -– Job Diagnostic Survey Hackman und Oldham (1975)	Indices für das 'Motivating Potential Score (MPS)': Anforderungsvielfalt, Ganzheitlichkeit der Aufgabe, Bedeutsamkeit der Aufgabe für andere, Autonomie, Rückmeldung	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): Vergleich von motivationspsychologisch relevanten Tätigkeitsmerkmalen
SAA -–Fragebogen zur subjektiven Arbeitsanalyse Udris und Alioth (1980)	A Handlungsspielraum B Transparenz C Verantwortung D Qualifikation E Soziale Struktur F Arbeitsbelastung	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): Vergleich von Belastungsstrukturen in der Arbeitstätigkeit
ISTA -– Instrument zur stressbezogenen Arbeitsanalyse (Fragebogenversion) Semmer (1984)	siehe ISTA (Beobachtungsversion)	siehe ISTA (Beobachtungsversion)

Siehe für weitergehende Informationen und vertiefte Darstellungen:

Baitsch C., Katz C., Spinas P. & Ulich E. (1989) Computerunterstützte Büroarbeit. Zürich: Verlag der Fachvereine.

Duell W. & Frei F. (1986; Hrsg.) Arbeit gestalten – Mitarbeiter beteiligen. Frankfurt: Campus.

Duell W. & Frei F. (1986) Leitfaden für qualifizierende Arbeitsgestaltung. Köln: TÜV Rheinland.

Duell W. & Katz C. (1990) Ratgeber Bildschirmarbeit. (Schriftenreihe der Bundesanstalt für Arbeitsschutz, Forschungsanwendung FA 24). Bremerhaven: Verlag für neue Wissenschaften.

Koch M., Reiterer H. & Gärtner J. (1990) EDV im Büro – Handbuch zur menschengerechten Gestaltung. Wien: Oldenbourg.

Frieling E. & Sonntag K. (1987) Lehrbuch Arbeitspsychologie. Bern: Huber.

Hacker W. (1986) Arbeitspsychologie. Bern: Huber.

Ulich E. (1991, 3. Auflage 1994) Arbeitspsychologie. Zürich: Verlag der Fachvereine, Stuttgart: Poeschel-Verlag.

Literatur

AMI (1992) A quantitative approach to software management. Centre for Systems and Software Engineering. South Bank University. 103 Borough Road, London SE1 0AA (UK).

ANDRIOLE S. (1989) Storyboard prototyping – a new approach to user requirements analysis. Wellesley: QED Information Sciences.

AUGUST J. H. (1991) Joint Application Design: the Group Session Approach to system design. London: Prentice Hall.

BAITSCH C., KATZ C., SPINAS P. & ULICH E. (1989) Computerunterstützte Büroarbeit. Zürich: Verlag der Fachvereine.

BALZERT H. (1986) Die Entwicklung von Software-Systemen: Prinzipien, Methoden, Sprachen, Werkzeuge. (Reihe Informatik, Bd. 34). Zürich: Bibliographisches Institut.

BALZERT H., HOPPE H. U., OPPERMANN R., PESCHKE H., ROHR G. & STREITZ N. A. (1988; Hrsg.) Einführung in die Software-Ergonomie. (Reihe Mensch Computer Kommunikation – Grundwissen 1; H. Balzert, Hrsg.). Berlin: de Gruyter.

BARTSCH-SPÖRL B. (1991) Erfahrungen aus Projekten, die nicht im Lehrbuch stehen. In: P. Elzer (Hrsg.) Multidimensionales Software-Projektmanagement. (S. 291–314). Hallbergmoos: AIT.

BECK A. & ZIEGLER J. (1991) Methoden und Werkzeuge für die frühen Phasen der Software-Entwicklung. In: D. Ackermann & E. Ulich (Hrsg.) Software-Ergonomie'91. (Berichte des German Chapter of the ACM, Band 33, S. 76–85). Stuttgart: Teubner.

BENZ C. & HAUBNER P. (1983) Gestaltung von Bildschirmmasken. *Office Management* 31, S. 36–39.

BOEDICKER D. (1990) Handbuch-Knigge – Software-Handbücher schreiben und beurteilen. (Angewandte Informatik, Band 6; H. Balzert, Hrsg.). Mannheim: BI Wissenschaftsverlag.

BOEHM B. (1981) Software Engineering Economics. Englewood Cliffs: Prentice Hall.

BOOCH G. (1994; 2nd ed.) Object-oriented analysis and design with applications. Redwood City: The Benjamin/Cummings Publ.

BORTZ J. (1984) Lehrbuch der empirischen Sozialforschung. Berlin: Springer.

BROWN C. M. (1989) Human-Computer Interface Design Guidelines. Norwood: Ablex.

CHIKOFSKY E. J. & RUBENSTEIN B. L. (1988) CASE: Reliability engineering for information systems. *IEEE Software* 5 (2), S. 11–17.

DEMARCO T. & LISTER T. (1991) Wien wartet auf dich! Der Faktor Mensch im DV-Management. München: Hanser.

DEMING W. E. (1989) Out of the crisis. Cambridge, Mass.: Massachusetts Institute of Technology

DIAPER D. (1989; ed.) Task Analysis for Human-Computer Interaction. New York: Ellis Horwood.

DIX A. J. (1991) Formal Methods for Interactive Systems. London: Academic Press.

DUELL W. & FREI F. (1986) Leitfaden für qualifizierende Arbeitsgestaltung. Köln: TÜV Rheinland.

DUELL W. & KATZ C. (1990) Ratgeber Bildschirmarbeit. (Schriftenreihe der Bundesanstalt für Arbeitsschutz, Forschungsanwendung FA 24). Bremerhaven: Verlag für neue Wissenschaften.

DUMAS J. S. & REDISH J. C. (1993) A practical guide to usability testing. Norwood: Ablex.

DUNCKEL H., VOLPERT W., ZÖLCH M., KREUTNER U., PLEISS C. & HENNES K. (1993) Das KABA-Verfahren. (Schriftenreihe Mensch-Technik-Organisation, Band 5; E. Ulich, Hrsg.). Stuttgart: Teubner.

DUTKE S. (1994) Mentale Modelle: Konstrukte des Wissens und Verstehens. (Arbeit und Technik, Band 4; M. Frese & H. Oberquelle, Hrsg.). Göttingen: Hogrefe.

EHN P. & KYNG M. (1991) Cardboard Computers: Mocking-it-up or Hands-on the Future. In: J. Greenbaum & M. Kyng (eds.) Design at Work: Cooperative Design of Computer Systems. (S. 169–195). Hillsdale: Lawrence Erlbaum.

ERICSSON K. A. & SIMON H. A. (1984) Protocol Analysis. Cambridge: Cambridge University Press.

FLOYD C. (1984) A systematic look at prototyping. In: R. Budde, K. Kuhlenkamp, L. Mathiassen & H. Züllighoven (Hrsg.) Approaches to Prototyping. (S. 1–18). Berlin: Springer.

FREEDMAN D. P. & WEINBERG G. M. (1990) Handbook of Walkthroughs, Inspections, and Technical Reviews. New York: Dorset House Publ.

FREI F. & UDRIS I. (1990; Hrsg.) Das Bild der Arbeit. Bern: Huber.

FRESE M. & BRODBECK F. (1989) Computer in Büro und Verwaltung. Berlin: Springer.

FRESE M., FRITZ A., STOLTE W. & BRODBECK F. (1991) Eine erste Analyse von Tätigkeiten und ihre Anforderungen. Posterbeitrag auf der Arbeitstagung "Software für die Arbeit von morgen". München.

FRIELING E. & HOYOS C. (1978) Fragebogen zur Arbeitsanalyse (FAA). Bern: Huber.

FRIELING E., KANNHEISER W., FACAOARU C., WÖCHERL H. & DÜRHOLT E. (1984) Entwicklung eines theoriegeleiteten, standardisierten, verhaltenswissenschaftlichen Verfahrens zur Tätigkeitsanalyse (TAI). (Institut für Psychologie). München: Universität.

FROSCHAUER U. & LUEGER M. (1992) Das qualitative Interview zur Analyse sozialer Systeme. Wien: Universitätsverlag der Hochschülerschaft Uni Wien.

FRÜHAUF K., LUDEWIG J. & SANDMAYR H. (1991a) Software-Projektmanagement und -Qualitätssicherung. Zürich: Verlag der Fachvereine.

FRÜHAUF K., LUDEWIG J. & SANDMAYR H. (1991b) Software-Prüfung – eine Fibel. Zürich: Verlag der Fachvereine.

GALITZ W. (1989) Handbook of Screen Format Design. P.O. Box 181, Wellesley, MA 02181 (USA).

GEMÜNDEN G. (1990) Erfolgsfaktoren des Projektmanagements - eine kritische Bestandsaufnahme der empirischen Untersuchungen. *Projekt Management* 1&2, S. 4–15.

GNIECH G. (1976) Störeffekte in psychologischen Experimenten. Stuttgart: Kohlhammer.

GREENBAUM J. & KYNG M. (1991, eds.) Design at Work: Cooperative Design of Computer Systems. Hillsdale: Lawrence Erlbaum.

HABERFELLNER R., NAGEL P., BECKER M., BÜCHEL A., & VON MASSOW H. (1992) Systems Engineering – Methodik und Praxis. Zürich: Industrielle Organisation.

HACKER W. (1989) Designing the designer's Tasks: Participative Analysis and Evaluation of Software Development Tasks. In: M. Smith & G. Salvendy (eds.) Work with Computers: Organizational, Management, Stress and Health Aspects. (p. 163–168). Elsevier: Amsterdam.

HACKER W., IWANOWA A. & RICHTER P. (1983) Tätigkeitsbewertungssystem TBS. Berlin: Humboldt Uni, Psychodiagnostisches Zentrum.

HACKMAN J. & OLDHAM G. (1975) Development of the job diagnostic survey. *Journal of Applied Psychology* 60, S. 159–170.

HAMPE-NETELER W. & RÖDIGER K-H. (1991) Software-Ergonomie: Verfahren der Evaluierung und Standards zur Entwicklung von Benutzungsoberflächen. (Fachbereich Mathematik/Informatik, Bericht Nr. 2/92). Bremen: Universität Bremen.

HEILMANN H. (1981). Modelle und Methoden der Benutzermitwirkung in Mensch-Computer-Systemen. Stuttgart: Forkel.

HELANDER M. (1991; ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

HESSE W., MERBETH G. & FRÖLICH R. (1992) Software-Entwicklung. (Handbuch der Informatik, Band 5.3). München: Oldenbourg.

HIX D. & SCHULMAN R. S. (1991) Human-Computer Interface Development Tools: a methodology for their evaluation. *Communications of the ACM* 34(3), p.75–87.

HOFFMANN T., KLOSE H.-G. & MARTIN H. (1989) Handbuch zur softwareergonomischen Gestaltung von Bildschirmmasken. (VDI Fortschrittsberichte Reihe 10: Informatik/Kommunikationstechnik Nr. 103) Düsseldorf: VDI Verlag.

JAMES B. (1989) The Systems Analysis Interview. Manchester: NCC Blackwell.

KIEBACK A., LICHTER H., SCHNEIDER-HUFSCHMIDT M. & ZÜLLIGHOVEN H. (1991) Prototyping in industriellen Software-Projekten – Er-

fahrungen und Analysen. (GMD-Studie Nr. 184). Sankt Augustin: GMD.

KISSLER L. (1988; Hrsg.) Computer und Beteiligung. Beiträge aus der empirischen Partizipationsforschung. Opladen: Westdeutscher Verlag.

KLOTZ U. (1991) Die zweite Ära der Informationstechnik. *Harvard Manager* 13(2), S. 101–112.

KOCH M., REITERER H. & GÄRTNER J. (1990) EDV im Büro – Handbuch zur menschengerechten Gestaltung. Wien: Oldenbourg.

KOSLOWSKI K. (1988) Unterstützung von partizipativer Systementwicklung durch Methoden des Softwareengineering. Opladen: Westdeutscher Verlag.

KUBICEK H. (1979) Interessenberücksichtigung beim Technikeinsatz im Büro- und Verwaltungsbereich. (Bericht Nr. 125 der Gesellschaft für Mathematik und Datenverarbeitung). München: Oldenbourg.

LAUTER B. (1987) Software-Ergonomie in der Praxis. Wien: Oldenbourg.

LEITNER K., VOLPERT W., GREINER B., WEBER W. & HENNES K. (1987) Analyse psychischer Belastung in der Arbeit (RHIA). Köln: TÜV Rheinland.

LIKERT R. (1972) Neue Aspekte der Unternehmensführung. (Schriftenreihe "Führung und Organisation der Unternehmung", Band 14). Bern: Paul Haupt.

MAASS S. (1994) Transparenz – eine zentrale Software-ergonomische Forderung. (Berichte des Fachbereiches Informatik FBI-HH-B-170/94). Hamburg: Universität Hamburg.

MAMBREY P. & OPPERMANN R. (1983; Hrsg.) Beteiligung von Betroffenen bei der Entwicklung von Informationssystemen. Frankfurt: Campus.

MAMBREY P., OPPERMANN R. & TEPPER A. (1986) Computer und Partizipation. Ergebnisse zu Gestaltungs- und Handlungspotentialen. Opladen: Westdeutscher Verlag.

MARTIN C. F. (1988) User-Centered Requirements Analysis. Englewood Cliffs: Prentice Hall.

MARTIN J. & MCCLURE C. (1985) Diagramming Techniques for Analysts and Programmers. Englewood Cliffs: Prentice-Hall.

MAUCH H. (1981) Metaplan. Nitor-Kommunikationstechnik GmbH, Jacksteinweg 10a, D-2000 Hamburg 52, Telefon +49-40-826 363.

MAYHEW D. J. (1992) Principles and Guidelines in Software User Interface Design. Englewood Cliffs: Prentice Hall.

MEYER-SCHÖNHERR M. (1992) Szenario-Technik. Ludwigsburg: Wissenschaft & Praxis.

MITTENECKER E. (1987) Video in der Psychologie. (Methoden der Psychologie, Band 9; K. Pawlik, Hrsg.). Bern: Huber.

MOLL T. (1987) Über Methoden zur Analyse und Evaluation interaktiver Computersysteme. In: K.-P. Fähnrich (Hrsg.) Software-Ergonomie. (State of the Art 5, German Chapter of the ACM, S. 179–190). München: Oldenbourg.

MULLER M. (1993) PICTIVE: Democratizing the Dynamics of the Design Session. In: D. Schuler & A. Namioka (eds.) Participatory Design: Principles and Practices. (S. 211–237). Hillsdale: Lawrence Erlbaum.

MÜLLER-HOLZ AUF DER HEIDE B., ASCHERSLEBEN G., HACKER S. & BARTSCH T. (1991) Methoden zur empirischen Bewertung der Benutzerfreundlichkeit von Bürosoftware im Rahmen von Prototyping. In: M. Frese, C. Kasten, C. Skarpelis & B. Zang-Scheucher (Hrsg.) Software für die Arbeit von morgen. (S. 409–420). Berlin: Springer.

MUMFORD E. & WELTER G. (1984) Benutzerbeteiligung bei der Entwicklung von Computersystemen. Berlin: Schmidt.

MUMMENDEY H.D. (1987) Die Fragebogen-Methode. Göttingen: Hogrefe.

MUSiC (1993) Metrics for Usability Standards in Computing. Teddington: National Physical Laboratory.

NIELSEN J. (1993) Usability Engineering. San Diego: Academic Press.

NIELSEN J. (1994, Ed.) Usability Laboratories. Special Issue of *Behaviour and Information Technology* 13(1–2).

NORMAN D. A. (1983) Design rules based on analyses of human error. *Communication of the ACM* 26(4), p. 254–258.

NORMAN D. A. (1989) Dinge des Alltags – Gutes Design und Psychologie für Gebrauchsgegenstände. Frankfurt: Campus.

OBERQUELLE H. (1987) Sprachkonzepte für Benutzergerechte Systeme. (Informatik-Fachberichte, Band 144). Berlin: Springer.

OPPERMANN R. (1983) Forschungsgegenstand und Perspektiven partizipativer Systementwicklung. München: Oldenbourg.

OPPERMANN R., MURCHNER B., REITERER H. & KOCH M. (1992; 2. Auflage) Software-ergonomische Evaluation: Der Leitfaden EVADIS II. Berlin: de Gruyter.

OTT H. J. (1991) Software-Systementwicklung: praxisorientierte Verfahren und Methoden. München: Hanser.

PESCHKE H. (1986) Betroffenenorientierte Systementwicklung. Frankfurt: Lang.

POMBERGER G. & BLASCHEK G. (1993) Grundlagen des Software-Engineering: Prototyping und objektorientierte Software-Entwicklung. München: Hanser.

RAASCH J. (1991) Systementwicklung mit strukturierten Methoden: ein Leitfaden für Praxis und Studium. München: Hanser.

RAUTERBERG M. (1991) Benutzungsorientierte Benchmarktests: eine Methode zur Benutzerbeteiligung bei der Entwicklung von Standardsoftware. (Projektberichte zum BOSS-Projekt Nr. 6; P. Spinas, M. Rauterberg, O. Strohm, D. Waeber & E. Ulich, Hrsg.). Zürich: ETH, Institut für Arbeitspsychologie.

RAUTERBERG M. (1993) AMME: an Automatic Mental Model Evaluation to analyze user behaviour traced in a finite, discrete state space. *Ergonomics* 36(11), p. 1369-1380.

RAUTERBERG M. & STROHM O. (1992) Work organization and software development. *Annual Review of Automatic Programming* 16(2), p. 121–128.

RICHARDSON S., DOHRENWEND B. & KLEIN D. (1965) Interviewing – Its Forms and Functions. New York: Basic Books.

RICHTER P., HEIMKE K. & MALESSA A. (1988) Tätigkeitspsychologische Bewertung und Gestaltung von Arbeitsaufgaben. *Zeitschrift für Arbeits- und Organisationspsychologie* 32, S. 13–21.

RÖDIGER K.-H., HAMPE-NETELER W. & PIEPENBURG U. (1991) Software-Ergonomie: Gestaltungsgrundsätze der DIN-Norm 66234 Teil 8 und ihre Umsetzung. Oberhausen: Technologie Beratungsstelle des DGB.

ROHMERT W. & LANDAU K. (1979) Das arbeitswissenschaftliche Erhebungsverfahren zur Tätigkeitsanalyse (AET). Bern: Huber.

RUCH L., SPINAS P. & ULICH E. (1989) Computerunterstützte Büroarbeit. Die Orientierung Nr. 95. Bern: Schweizerische Volksbank.

RUDOLPH E., SCHÖNFELDER E. & HACKER W. (1987) Tätigkeitsbewertungssystem – geistige Arbeit. Berlin: Humboldt Uni, Psychodiagnostisches Zentrum.

RUPIETTA W. (1987) Benutzerdokumentation für Softwareprodukte. (Angewandte Informatik, Band 3), Mannheim: B.I.

SCHIEMENZ B. (1979) Kybernetik. In: W. Kern (Hrsg.) Handwörterbuch der Produktionswissenschaft. (S. 1022–1028). Stuttgart: Poeschel.

SCHREIBER J. (1994) Beschaffung von Informatikmitteln: Pflichtenheft – Evaluation – Entscheidung. Bern: Schweizerische Vereinigung für Datenverarbeitung.

SCHÜPBACH H. (1993) Analyse und Bewertung von Arbeitstätigkeiten. In: H. Schuler (Hrsg.) Lehrbuch Organisationspsychologie. (S. 167–187). Bern: Huber.

SEMMER N. (1984). Stressbezogene Tätigkeitsanalyse. Weinheim: Beltz.

SOMMERVILLE I. (1989) Software Engineering. New York: Addison-Wesley.

SPINAS P., TROY N. & ULICH E. (1983) Leitfaden zur Einführung und Gestaltung von Arbeit mit Bildschirmsystemen. München: CW.

STEWART C. (1983) Interviewing: Principles and Practices, a Project Text. Dubuque: Kendall/Hunt.

STREITZ N. (1990) Psychologische Aspekte der Mensch-Computer-Interaktion. In: C. Hoyos & B. Zimolong (Hrsg.) Ingenieurspsychologie. (Enzyklopädie der Psychologie, Themenbereich D – Praxisgebiete, Serie III, Wirtschafts-, Organisations- und Arbeitspsychologie, Band 2, S. 240-284). Göttingen: Hogrefe.

STROHM O. & ULICH E. (1991) Arbeitsteilung und Benutzerorientierung bei der Software-Entwicklung. In: P. Elzer (Hrsg.) Multidimensionales Software-Projektmanagement.(S. 261–289). Hallbergmoos: AIT.

TROY N., BAITSCH C. & KATZ C. (1986) Bürocomputer – Chance für die Organisationsgestaltung? (Arbeitswelt, Band 3, A. Alioth, Hrsg.). Zürich: Verlag der Fachvereine.

UDRIS I. & ALIOTH A. (1980) Fragebogen zur subjektiven Tätigkeitsanalyse (SAA). In: E. Martin, I. Udris, U. Ackermann & K. Oegerli (Hrsg.) Monotonie in der Industrie. (S. 61–68). Bern: Huber.

ULICH E. (1985) Einige Anmerkungen zur Software-Psychologie. *Sysdata* 16(10), S. 53–58.

ULICH E. (1986a) Aspekte der Benutzerfreundlichkeit. In: W. Remmele & M. Sommer (Hrsg.) Arbeitsplätze von morgen. (Berichte des German Chapter of the ACM, Band 27, S. 102-121). Stuttgart: Teubner.

ULICH E. (1986b) Neuorientierung des Managements. In: W. Duell & F. Frei (Hrsg.) Arbeit gestalten – Mitarbeiter beteiligen. Eine Heuristik qualifizierender Arbeitsgestaltung. Frankfurt: Campus, S. 160–169.

ULICH E. (1989) Arbeitspsychologische Konzepte der Aufgabengestaltung. In: S. Maass & H. Oberquelle (Hrsg.) Software-Ergonomie'89. (Berichte des German Chapter of the ACM, Band 32, S. 51–65). Stuttgart: Teubner.

ULICH E. (1991) Gruppenarbeit – arbeitspsychologische Konzepte und Beispiele. In J. Friedrich & K.H. Rödiger (Hrsg.) Computergestützte Gruppenarbeit (CSCW). (Berichte des German Chapter of the ACM, Band 34, S. 57–77). Stuttgart: Teubner.

ULICH E. (1994; 3.Auflage) Arbeitspsychologie. Zürich: vdf Hochschulverlag AG.

VÖGELE S. (1992) Dialogmethode – das Verkaufsgespräch per Brief und Antwortkarte. Zürich: Moderne Industrie.

VOLPERT W. (1987) Psychische Regulation von Arbeitstätigkeiten. In: U. Kleinbeck & J. Rutenfranz (Hrsg.), Arbeitspsychologie. (Enzyklopädie der Psychologie, Themenbereich D, Serie III, Band 1, S. 1–42), Göttingen: Hogrefe

VOLPERT W., OESTERREICHER R., GABLENZ-KOLAKOVIC S., KROGOLL T. & RESCH M. (1983) Verfahren zur Ermittlung von Regulationserfordernissen in der Arbeitstätigkeit (VERA). Köln: TÜV Rheinland.

WALLMÜLLER E. (1990) Software-Qualitätssicherung in der Praxis. München: Hanser.

WANDKE H. (1988) Mensch-Rechner-Interaktion: Optimierung des Informationsaustauschs. *wissenschaft und fortschritt* 40(7), S. 173–177.

WANDMACHER J. (1993) Software-Ergonomie. (Mensch-Computer-Kommunikation: Grundwissen, Band 2), Berlin New York: de Gruyter.

WICKE W. (1988) Methoden der Partizipation bei der Entwicklung computergestützter Arbeitssysteme. In: L. Kissler (Hrsg.) Computer und Beteiligung. Beiträge aus der empirischen Partizipationsforschung. (S. 117–140). Opladen: Westdeutscher Verlag.

WOOD J. & SILVER D. (1989) Joint Application Design – How to Design Quality Systems in 40% Less Time. New York: Wiley & Sons.

ZAPF D. & FRESE M. (1989) Benutzerfehler im Kontext von Arbeitsaufgabe und Arbeitsorganisation. In: S. Maass & H. Oberquelle (Hrsg.) Software-Ergonomie'89. (Berichte des German Chapter of the ACM, Band 32, S. 213–222). Stuttgart: Teubner.

ZIEGLER J. & ILG J. (1993; Hrsg.) Benutzergerechte Software-Gestaltung. München: Oldenbourg.

ZÖLCH M. & DUNCKEL H. (1991) Erste Ergebnisse des Einsatzes der "Kontrastiven Aufgabenanalyse". In: D. Ackermann & E. Ulich (Hrsg.) Software-Ergonomie '91. (Berichte des German Chapter of the ACM, Band 33, S. 363–372). Stuttgart: Teubner.

Personenverzeichnis

Alioth 9, 201
Andriole 125
August 86, 115f.
Baitsch 9, 130, 197, 201
Balzert 197
Bartsch-Spörl 73, 192
Beck 113
Benz 17
Blaschek 126
Boedicker 21
Boehm 86, 90
Booch 134
Bortz 136
Brown 197
Chikofsky 88
DeMarco 77, 80
Deming 84
Diaper 86
Dimon 137
Dix 125
Dohrenwend 119
Duell 130, 197, 201
Dumas 93, 133, 138
Dunckel 86, 200
Dutke 145
Dzida 15f.
Ehn 123, 134
Ericsson 137
Floyd 126
Freedman 131, 133
Frei 201
Frese 18, 75
Frieling 200f.
Frölich 82, 88
Froschauer 119
Frühauf 81, 87f., 131, 133
Galitz 197
Gärtner 130, 201
Gemünden 75
Greenbaum 94
Haaks 24
Haberfellner 94
Hacker 75, 199, 201
Hackman 201
Hampe-Neteler 197, 208
Haubner 17
Heilmann 94
Helander 13
Hesse 82, 88
Hix 129
Hoffmann 197
Hoppe 197, 202
Hoyos 200
Ilg 15, 198, 211
James 119
Katz 130, 197, 201
Kieback 127
Kissler 2, 27
Klein 119
Klose 197, 205
Klotz 85
Koch 130, 201
Koslowski 2, 27
Kubicek 63
Kyng 94, 123, 134
Landau 200

Lauter 197
Likert 78
Lister 77, 80
Ludewig 81, 87f., 131
Lueger 119
Maass 16
Mambrey 1, 27, 42
Martin 86, 125, 197, 205
Mauch 116
Mayhew 197
McClure 86, 125
Merbeth 82, 88
Meyer-Schšnherr 124
Mittenecker 138
Moll 137
Mollenhauer VIII
Mosier 194
Muller 124
MŸller-Holz 88
Mumford 1, 27
Mummendey 119, 121
Nielsen 93, 133
Norman 18, 19
Oberquelle 86, 125
Oldham 201
Oppermann 1, 27, 42, 131f., 197, 202
Ott 87
Peschke 2, 197, 202
Pomberger 126
PrŸmper 196
Raasch 86, 125
Rauterberg 127, 139
Redish 93, 133, 138
Reiterer 15, 130, 201
Richardson 119
Richter 199
Ršdiger 197
Rohmert 200
Rohr 197, 202
Rubenstein 88
Ruch 2
Rupietta 21
Sandmayr 81, 87f., 131
Schiemenz 83
Schlagenhauf VIII
Schreiber 122
Schulman 129
SchŸpbach 199
Semmer 200
Silver 86, 115f.
Smith 194
Sommerville 87
Sonntag 201
Spinas 2, 130, 197, 201
Stewart 119
Streitz 13, 197, 202
Strohm 75, 127
Tepper 2, 27
Troy 197, 209
Udris 201
Ulich 2, 5, 7, 9Ð11, 15f., 20, 60, 75f., 80, 197, 201, 209
Všgele 121
Volpert 63, 199
WallmŸller 64
Wandke 17, 196
Wandmacher 197
Weinberg 131, 133
Welter 1, 27
Wicke 94
Wood 86, 115f.
Zapf 18
Ziegler 113, 198

Rauterberg, M., Spinas, P., Strohm, O., Ulich, E. & Waeber, D. (1994).
Benutzerorientierte Software-Entwicklung Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung.
(Mensch - Technik - Organisation. Band 3), vdf Hochschulverlag AG an der ETH Zurich.

Stichwortverzeichnis

A
Ablauf 41, 115f., 118, 131, 165, 182
Ablauforganisation 4, 40, 53f., 67, 81, 107, 152, 155, 172, 179
ADI VIII, 167, 169, 170, 194f.
ADIMENS 167Ð171
Akzeptanz 1, 2, 17, 23, 27, 29, 47, 178
Alpha-Test 88
AMI 82f.
AMME 139
Analyse 94, 96, 128
  Arbeitsablauf 181
  Arbeitsfaktoren 199
  Arbeitsgruppe 68
  aufwand 86
  Benutzerbeteiligung 104
  Datenstruktur 72
  der Aufgabenstruktur 113f.
  der Datenstruktur 101, 113f.
  der Fehler 146
  des Arbeitssystems 85f., 90, 97, 106f.
  -ergebnis 107
  funktionsbezogen 184
  innovative Projekte 58
  Ist-Analyse 177
  Meta-Analyse 75
  Methoden 41, 73, 94, 125, 161
  -phase 75, 85Ð87, 90, 94, 100, 112
    und kommunikative Methoden 86
  Projekte VII, 90
  Projektgruppe 72
  Projektrahmenbedingungen 56
  Prozessmodellierung 85
  Prozessmoderation 85
  quantitativ 31
  Schwachstellen 133f.
  soziotechnische 107
  suboptimale 86
  und Anforderungsermittlung 186
  -verfahren 129, 199
  Video 140, 194
  von grossen Projekten 75
  von Problemen 115, 151, 157, 181
  wirtschaftliche 181
  Zeitreserven 187
Analytiker VII, 4, 72, 75, 128, 184
  als Projektleiter 156
  Programmierer 75f., 130, 163
  Projektleitung 150
Anforderungen 10, 50, 56, 65f., 85, 108
  nderungen 3, 30, 50, 74, 97
  Arbeitsablauf 1
  Arbeitssystem 110
  Aufgaben 112
  Aufgabengestaltung 111
  Aufgabenstruktur 114
  aus Schwachstellen 109
  Benutzer VII, 1, 50
  Benutzergruppe 151
  BenutzungsoberflŠche 111
  Benutzungstest 140
  Datenstruktur 112
  der Organisation 9
  Entwurf 193
  Ermittlung 200
  Evaluation 25
  Formulierung 109
  funktionelle 169
  Gewichtung 109
  kognitive 114, 199
  komplexe 161
  LŠsungsvarianten 110
  Mensch-Computer-Funktionsteilung 111f.
  Pflichtenheft 122
  Qualifikation 6
  QualitŠt 107
  realistisches Systemverhalten 140
  Reviews 77
  Schulung 29, 189
  Software-Entwicklung 171
  softwareergonomische 112, 115, 126
  Soll-Konzept 182
  Spezifikation 81
  systembedingt 177
  technische 3, 112
  und Hintergrundwissen 105
  vergessene 150
  widerspruchliche 166
Anforderungs
  -analyse 31, 65, 68, 81, 91, 93, 108, 112
  -definition 105f., 109, 161
  -ermittlung 31, 42, 64, 112, 114, 151, 186f.
    Prototyping 161

Rauterberg, M., Spinas, P., Strohm, O., Ulich, E. & Waeber, D. (1994).
Benutzerorientierte Software-Entwicklung Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung.
(Mensch - Technik - Organisation. Band 3), vdf Hochschulverlag AG an der ETH Zurich.

-höhe einer Teiltätigkeit 114
-katalog 25
-konfiguration 200
-vielfalt 10f., 15, 201
Ängste 28, 141
Animation Works 194
Antwortzeit 16, 126, 179f.
Anwendung VII, 2, 19, 35, 37, 39, 112, 114, 192f.
Anwendungsspezialist 69
Apple 195
Arbeitnehmervertretung 95
Arbeits
-analyse 159
-aufgaben 95
-gestaltung 26, 29, 105, 201
-gruppe 44, 62, 64, 68f., 154, 176, 177, 186–188
Anleitung von 118
Arbeitsaufträge 68
auf operativer Ebene 72
aufgabenspezifische 68
Auftragszuteilung 70
bereichsspezifische 176
betriebsinterne Anwendung 39
Ergebnispräsentation 116, 165
gemischte 54
Grösse 68
Hierarchiegefälle 47
interessenspezifische 67
Koordination 68
koordinierte 68–70
Moderator 166
parallel 66–68, 70, 110
Schulung 144
selbständige 68
Teilnehmerzahl 116
themenspezifische 54, 165
und Spezialisten 69
Verantwortung 69
Verbindungsglieder 68
Vertreter 68
zusätzliche 66
-inhalt 6, 28
-organisation 59, 74, 104f., 111, 126, 184, 188
-system 7
-tätigkeit 1, 5–7, 178, 199, 201
Belastung 200
geistige 199–201
industrielle 200
Klassifikation 200
Regulationshindernisse 200
-teilung 59f., 177, 179
funktionale 60
Nachteile 62, 75
produktorientierte 77
-wissenschaftler 64, 66, 69
Astound 194
Aufbauorganisation 60, 61f., 70, 79, 90
alternative 63
des Projektes 53f., 60, 176
des Softwarehauses 171
Form der Beteiligung 39
operative Ebene 53
traditionelle 61
Varianten 64
Aufgabe 185
der Arbeitsgruppe 187
der Benutzer 188
der Benutzervertreter 188
erweiterte 181
Ganzheitlichkeit 201
gemeinsame 181
realistische 190
und Arbeitsabläufe 186, 189
Aufgaben
-analyse 113, 200
-orientierung 111f.
angemessenheit 15, 27, 112, 146, 152, 164
gegenstand 112
kontext 26, 57, 70, 88
merkmale 113, 199
struktur 113f.
verteilung 187
zusammenhang 9, 76
Aufmerksamkeit 12, 14, 97
Ausbildung 2f., 29, 36, 37, 49, 62, 99, 101, 108, 118, 136, 141–145, 154, 164, 178f., 185, 187–189
Ausbildungsinhalte 143, 189
Autonomie 10, 11, 15, 201

B

Beeinträchtigungslosigkeit 5, 199f.
Befragung 119
schriftlich 120
Benutzer 57
als Programmierer 37
unzufriedene 105
Benutzerbeteiligung 26, 36, 41, 46, 50, 95, 99, 101, 103f., 172, 187
und Transparenz 95

Benutzergruppe 58, 99, 101, 103, 151, 154, 156, 185
aktive Mitarbeit 154
repräsentative 154
und Fluktuation 155
und Mitbestimmungsrechte 151
und Workshop 102
Zusammensetzung 100, 136
Benutzermerkmale 99
Benutzertypen 41
Benutzervertreter 38, 42, 45–48, 62, 66, 74, 76
Auswahl 46
Benutzung 96
Benutzungsfreundlichkeit 130
Benutzungsoberfläche VII, 1, 13, 25, 26, 40, 44, 112, 115, 123, 126, 158f., 161, 163f., 167, 184, 196, 198, 205
Bewertung 131
Desktop 160
direktmanipulierbare 17, 19, 167
formale Gestaltung 31
grafische 37
Hilfesysteme 20
individuelle Anpassung 23
Interaktionsformen 19
Mängel der 146
multimediale 20
Transparenz 16
und Interaktionsformen 24
und Metaphern 20
und Prototyping 126, 127
Benutzungstest 88, 92, 93f., 133, 137, 169, 170
aufgabenorientiert 134
induktiv 134
Beobachtung 94
Berater
des Softwarelieferanten 175
externe 151, 178, 186
interne 150
Beta-Test 88, 170f.
Beteiligung VII, 3f., 15, 26f., 29, 30, 40, 43–45, 50, 97, 101f., 156, 158
aktive 31, 44, 106
als Alibiübung 186
als Lernprozess 50
am Review 131
Bedenken des Projektleiters 156
der Fachbereiche 181
direkte 38, 100, 104, 113, 171
evaluative 42f.
indirekte 38, 100
Inhalte 100
Nachteile 43
Notwendigkeit 30
nutzlose 46
passive 42
Praxisprobleme 46
Probleme 129
Projektphasen 84
prozessuale 44
Rahmenbedingungen 38
repräsentative 185
und Benutzertagung 155
und Betroffenenstruktur 98
und Code-Inspektion 133
und externe Berater 151
und Information 159
und innovatives Vorgehen 184
und Probleme 95
und Projektleitung 156
und Repräsentativität 40
und Verweigerung 95
Vorteile 27
Beteiligungs
-aktivitäten 100f.
-form 42
-konzept 50
-methoden 93, 133, 188
-modell 39, 46, 95, 97–101, 103, 159
Grobstruktur 96
schriftliche Fassung 96
und Betroffenenstruktur 98
und Merkmale des Arbeitssystems 97
und personelle Besetzung 98
und Projektrahmenbedingungen 98
und Zeitrahmen 98
und Zielsetzungen 98
-organisation 95
-prozess 102, 104
-wunsch 102
Betriebsrat 33, 45, 186
Betriebsverfassungsgesetz 33
Bewertung 94
des Arbeitssystems 97
BOSS 167
Brainstorming 94, 124
Bürokratisierung 62, 78

C

CASE 88, 192
CD-ROM 165
Checkliste 87, 130
AWV 197
EDV 197

Software Checker 196
Codierer 184
Codierung 81
Comic strips 94
Computererfahrung 99
Controlling 81
critical incidents 139
CUA 195
CUI 167

D
Darstellung 73, 94
data dictionary 88
Daten
-modell 85, 102
-schutz 94
-struktur 66, 72, 100–102, 113f.
Datenbank 13, 32, 98, 102, 123, 125, 131, 160, 166f., 171, 192
und Marketing 171
zur Versionenverwaltung 93
Datenbankkern 168f.
Design 92, 197, 207
Guidelines 194
studien 126
vorschläge 128
Designer 4
Dialogablauf 22f., 40
Dialogsteuerung 164
Dialogstruktur 18, 23, 25, 155
DIN-66234 15f., 197, 208
DIN-66234 172
Dokumentenanalyse 94
Doppelbelastung 46
Drehbuch 125
DRUID 194
DRUM 194

E
EDV 43, 142, 151, 189, 200
-Abteilung 163, 184
-Qualifikation 151
-technische Fragen 153
-technische Unterstützung 158
-Vertreter 158, 192
EDV-Erfahrung 101
EDV-Unterstützung 95
Effizienz 105f.
EG-Richtlinie 15
Einführung 28, 33, 64, 104, 144, 174f., 178–181, 190
des Systems 189
standardisiert 137
und Information 177
und Nachevaluation 146
und Schulung 145, 178
und Übungsarbeitsplatz 144
Verzögerung 36
von PPS 178f.
Einführungs
-phase 137, 145
-prozess 1
-termin 193
-versuch 30
email 121
Empathie 74
Endprodukt 25, 29
Entscheidungskompetenz 75
Entwicklungsmittel 53f.
Entwicklungsmöglichkeiten 15
Entwicklungsumgebung 54, 87
und Produktivitätssteigerung 88
ETH 167
EVADIS 131f.
Evaluation
des EDV-Systems 190
gemeinsame 71
heuristische 92
in der Benutzergruppe 100f.
nachträglich 146f.
und Benutzungstest 134
und Beteiligung 100f.
und Methodenkompetenz 73
und Probleme 128
von Arbeitsstrukturen 182
von CASE-Werkzeugen 192
von Prototypen 126, 128f., 133
von Veränderungsprozessen 200
von Zwischenstadien 25
Evaluations
-kriterien 115
-methoden
Kritik 198, 205
-technik 131
-zyklen 128
Experte 28, 108, 145, 153, 168, 191, 193
Benutzer als VII, 2
Entwickler als VII, 2
externer 38
lokaler 145
Nachteile 29
Vorteile 145
Experten
-evaluation 86
Kooperation 40
Modus 152
Expertensystem 70

F
Fachbereich 41, 55, 57f., 64–68, 70, 73f., 176, 178, 181
Integration 64
Kadervertreter 66
Mitarbeit des 68

Fachbereichsvertreter 66, 68, 70f., 175, 177, 181
Fachbereichsvertretung 79
Fachkompetenz 30, 73, 75
Fachwissen 2, 29, 58, 135, 185, 188
Feedback 15–18, 27, 32, 35, 42, 44, 48, 66, 68, 79, 151, 158, 166
  über das Produkt 39
Fehler
  denklogischer 180
Fehlerbehebungskosten 86
Feinkonzept 85, 182
Feldstudien 88
finanzielle Probleme 97
Fixierung 47, 190
Flexibilität 15, 98
  der Computerinteraktion 22
Flipchart 92, 116
Fluktuation 46, 75, 97
Form der Beteiligung 39, 66, 187
Formulargestaltung 68
Fragebogen 39, 43, 88, 94, 101, 119
  aktion 102
  doppelte Verneinung 120
  Formulierungen 120
  Gründe für Nichtausfüllen 121
  indirekte Fragen 119
  interaktiver 121f.
  Kurzform 170, 172
  Mahnwesen 120
  Registrierkarte 94
  Rücklaufquote 120
  und Arbeitsanalyse 201
  und Benutzermerkmale 99
  und evaluative Beteiligung 42
  und Fachzeitschriften 169
  und Maskendesign 165
  Vorteile 120
  zur Messung der EDV-Kenntnisse 138
  zur Workshopvorbereitung 165
Freistellung 96, 104
Funktionalität 30, 59, 66, 111, 126, 128, 153, 160f., 163, 168f., 192f.
  eines Prototyps 159f.
Funktionsteilung
  Mensch-Computer 1, 86, 104, 111f., 159, 184
  Mensch-Mensch 1

G

Ganzheitlichkeit 15, 132, 201
Gerüchte 29, 33, 187
Geschäftsvorfall 150
Gesprächsführung 37
Gestaltung
  des Arbeitssystems 97
Gestaltungsziele 106, 108, 110
globale Daten 87
Grad der Beteiligung 40, 187
Grenzbedingungen 71
Grenzregulation 9, 72
Grobkonzept 42, 85
Gruppenleistungsanteile 79
GUI 167
Guidelines 195
  Smith & Mosier 196

H

Handskizze 86, 92, 123, 125
Hardware 168
  -gestaltung 105
  -kosten 178
  mangelhafte 82
  preiswerte 14
  und Betriebssystem 98
  und Entwicklung 7
  und Ergonomie 1
Hierarchiegefälle 47
HIPO 94, 125
Hypercard 194

I

IBM 194, 195
IconAuthor 194
Identifikation 2, 27–29, 70, 73, 78, 106
Implementation 64, 96
individuelle Auswahl 15
Information 114, 120f., 139, 145, 150, 152, 157–160, 187f.
  Auswertung der 171
  Darstellung 160
  der Projektgruppe 158
  im Dialog 155
  in Seminaren 187
  Präsentation 160
  über die Hauszeitung 187
  Verbreitung der 187
  Verdichtung von 160
Informations
  -abfrage 154
  -aufnahme 200
  -bearbeitung 114
  -darstellung 115
  -erarbeitung 114
  -fluss 177
  -fortlaufende 188
  -gewinnung 157
  -organisation 152f.
  -politik 96, 104, 122
  -suche 151
  -system 163
  -übertragung 114
  -veranstaltungen 159, 182
  -verarbeitung 114

-verteilungsnetz 187
Inhalt 187
der Beteiligung 40
Inkonsistenzen 18, 50, 91
Innovation 2, 9, 28
ganzheitliche Sichtweise 173
organisatorische 181f.
technologische 107, 182
und Ungewissheit 29
Innovationsvorhaben 28
Installation 81
Integration 81
Interaktion 3, 7, 100
flexible 17
Mensch-Computer 16
mit dem EDV-System 152
O-F-Struktur 19
soziale 10f., 15
Steuerung der 13
Interaktions
-formen 24
-möglichkeiten 152
-problem 13, 18, 25
-schnittstelle 13
Interessenkonflikt 1
Interessenvertretung 97
Interface Design 196, 197, 203
Interface Ratgeber 196
Interview 88, 94, 119
telefonisch 120
ISO-9004 172
ISO-9241 15f.
ISO-Fragebogen 196
Iteration 3
Iterationsschritt 193
iterativ-zyklisch 54, 56, 81, 93, 100f., 112, 127, 158, 160
JACKSON 125

K
Kärtchenmethode 94
Kenntnisse 10
EDV 19, 24, 29, 36, 37f., 49, 141f., 189
fachspezifische 29, 37, 42, 162
funktionale 142, 145
in Gesprächsführung 37
operative 142, 145
Schulung 143
softwareergonomische 37
und Schulung 143
Kommunikation 108, 186, 199f.
Entwickler-Benutzer 100, 128, 161f., 168
lautes Denken 137
soziale 73, 77
verbale 92
Kommunikations
-erfordernisse 58
-kanäle 187
-probleme 49, 186
-prozess 180
-wege 110
Kompatibilität 15f., 152
Konfliktbewältigung 186
Konsistenz 15f.
Kontinuität 75
Kooperation 10, 27, 34, 40, 63, 77, 82, 95, 100, 108, 158, 165, 199
Kooperationserfordernisse 58
Kooperationsmöglichkeiten 75
Kooperationspartner 97
Kooperationsstruktur 108
Koordinator VII, 37, 45, 145, 151
Kreativität 116
Kriterien 7, 17, 25
arbeitspsychologische 4, 16, 26
für Beteiligung 40
für menschengerechte Arbeit 5
für Projektmanagement 52
softwareergonomische 22, 26, 31, 98, 150, 163, 167
softwaretechnische 14, 87, 114, 126, 167
und Anforderungsgewichtung 109
und Arbeitsabläufe 125
und Benutzerbeteiligung 26
und Benutzungstest 140
und Projekterfolg 52
von EVADIS 132
zur Bewertung der Aufbauorganisation 60
zur Erfolgsmessung 51

L
Lenkungsausschuss 39, 61–64, 66, 69, 72
Lernen 143
exploratorisches 145
in Kleingruppen 143
mit Aufgabenbezug 143
Lernmöglichkeiten 15
Logfile 88, 94

M
Machtpromotor 74
MacroMind Director 195
Mailbox 166
Management 1, 61, 158, 186
Informationssystem 192
Marketing 150f., 172

Maskendesign 40, 44f., 68, 86, 154, 164, 197, 204–206
Maskengenerator 88, 94
Meilensteine 81
Metaplan 92, 116
Methoden
der Beteiligung 41
informale 90
kommunikative 90
semiformale 90
Methodenkompetenz 73
Mitarbeiterstruktur 108
Mitwirkung 4, 29, 38, 60
mock-up 125
Moderation
durch externen Moderator 109
von Arbeitsgruppen 165
von Gruppen 37
von Workshops 115
Moderator 109, 116–118, 166, 191
als neutrale Person 118
eines Workshops 116f.
externer 109
Moderatoren
-schulung 118
-rolle 71, 118
-schulung 115
Modultest 81
Motivation 11, 48
innere 11
mangelnde 74f., 104, 178f.9, 184
und Arbeitsablauf 124
und Schulung 145
zur Beteiligung 43, 46, 50, 75, 78, 106, 118, 137, 160
zur Systemnutzung 27, 29
Motivationserhalt 48
Multimedia 20
MUSiC 140, 196

N
Netzwerk 98
Neuentwicklung 57
NeXT 195
Nichtbeteiligte 96, 104
Normen 25

O
Optimierungsaufwand 87, 90
Optimierungszyklus 83, 89, 91
effizienter 84
global 90
lokal 90
und Störgrössen 84
und Zykluszeit 84
Organisation 4, 37, 150, 177
als soziotechnisches System 8
Bedürfnisse der 108
der Arbeit 76
der Beteiligung 185, 187
des EDV-Systems 16
formale Struktur 107
lebendige 85
mit EDV 125
unabhängige Einheiten 9, 62
und Arbeitsaufgabe 11
und Ausbildung 188
und EDV-Einsatz 9
und Heterogenität 150
und Informatik 36
und Lebensfähigkeit 85
und Mitwirkung 29
und Produkt 9
und Qualitätssicherung 193
von Workshops 44
Organisations
-analysen 111
-defizite 85
-einheit 9, 107, 143, 155
-entwicklung 4, 106
-fachleute 105
-form 4, 39, 44, 93, 155
-gestaltung 80, 93, 182
-konzept 182
-modell 63, 70, 75
-modelle 80
-psychologen 167
-psychologie 86, 90
-schnittstelle 85
-struktur 7, 80, 95, 110f., 174f., 181
-ziele 85, 155
OSF/Motif 196
Overhead 116

P
Paper pencil 94
Papier und Bleistift 125
Partizipation 2, 4, 15, 29, 75, 92, 167f., 172, 188
Angebot 169
Bereitschaft 169
pseudo 29, 185
Patentrezept 46
pay for knowledge 79
persönlichkeitsförderlich 5f., 199
Petri-Netze 125
Pflichtenheft 122, 186
Phasenmodell 81, 158, 163
Nachteile 82
notwendige Voraussetzungen 83
und Projektmanagement 81
und Steuerung 82
Pinwand 116
Praxis 91
-beispiele 149
betriebliche 75

-erfahrungen 27, 29
falsche Entwicklungsrichtung 7
-test 39
und Anforderungsanalyse 112
und Ausbildung 189
und Benutzungsoberflächen 24
und Form der Beteiligung 39
und Fragebögen 121
und Information 35
und Schulung 141
und Softwareergonomie 197, 206
und Übungsarbeitsplatz 144
Primäraufgabe 9
Produkt 3, 9, 27, 76, 107, 167, 172
-datenbank 171
-einsatz 171
-entwicklung 167, 169
fehlerfrei 126
-festlegung 173
-güterindustrie 126
-qualität 61, 91, 172
-segmentierung 181
und Entwickler 118
und Entwicklungsumgebung 88
und Prototyp 161
und Registrierkarte 39, 170
unfertig 126
varianten 167
verantwortlicher 133, 170, 172
verbreitung 167
versionen 39
Weiterentwicklung 170
Produktion 41, 73, 77, 94, 127, 162
von Prototypen 127
Produktions
-konzept 181f.
-planung 57, 174
-prozess 107
Produktivität 1, 75
Produktivitätssteigerung 152
Programmgenerator 88
Programmierer VII, 4, 21, 44f., 72, 75f., 130, 150, 163, 184
Kooperation 75
unzufriendene 75
Projekt VII, VIII, 1, 26, 34f., 40, 50, 56, 59
-abbruch 56
-ablauf 38, 42
-abmachungen 97
-abwicklung 53
-aktivitäten 51
-arbeit 31, 38, 48, 54, 64
-aufgabe
vollständig 76
-aufgaben 53
aus dem Zeitplan 74
-beginn 41, 50, 55
-beschreibung 60
-beteiligte 37f., 51, 53, 57, 60f.
Branchenlösung 56
-broschüre 35
-budget
zu klein 74
-dauer 47, 57
-einheit 62
-ende 47
-erfolg 51f., 55
Bedingungen 79
Kriterien 75
Warnsignale 74
-etablierung 91
fehlende Werkzeuge 74
fehlendende Qualifikation 74
Fluktuation 46
-fortschritt 47
-gestaltung 55
-gremium 64
grosses 62f.
-grösse 34f., 55
-gruppe 44, 47, 53f., 62–64, 68, 74, 154, 158
betriebsinterne Anwendung 39
gemischte 54, 66, 72
-gruppen 63, 75–78
Aufgabenzusammenhang 76
gemischte 64
Grösse 79
Transparenz 79
und Bezahlung 79
und Fluktuation 78
und Prototyping 79
Vorteile 78
-handbücher 3
im betrieblichen Kontext 34
-inhalt 40
innovatives 54, 58
innovativität 57
keine Qualitätssicherung 74
-klassifikation 53f., 56, 60, 149
komplexes 57
-komplexität 57
unterschätzt 74
-kontrolle 53
-koordinator 68
-laufzeit 50
-leiter VII, 2, 4, 33, 37, 45f., 63, 68, 71f., 75
als Vermittler 71
betriebspolitische Funktion 72
Grenzregulation 83
Koordinationsaufgaben 77

technischer 95
und Beteiligte 73
und Empathie 73
und Fachkompetenz 73
und Methodenkompetenz 73
und Planungswissen 82
und Sozialkompetenz 73
und Warnsignale 74
-leitung 63f., 70f.
-management 2, 51, 52–54, 56, 62f.
mangelhafte Ergebnisdokumentation 74
mangelnde Motivation 74
massgeschneidert 46
Misserfolg 29, 34
-nachrichten 35
-organisation 62, 66
-phasen 62
-planung 58
-rahmenbedingungen 56, 97
-schritte 60
sinkendes Engagement 46
-skizze 35
strategische Schritte 53
-struktur 63
-typ 39, 56f., 64
und Quadrantenmodell 89
und Softwareentwicklung 89
überschätzt 74
und Arbeitsteilung 60
und Auskunft 36
und Auswahl der Beteiligten 41
und Beteiligung 38, 42–44, 46f., 50
und Gerüchte 33
und Kontrolle 63
und Meilensteine 45
und Methodeneinsatz 41
und Motivation 48
und Probleme 50
und Teilprojekte 48
ungewolltes 74
-verantwortlicher 36
-verlauf 51
-vorhaben 60
-ziele 53, 55, 60, 64, 71, 97
unklar 74
zu starrer Softwareentwurf 74
Projektetablierung 95
Prototyp 160–162, 166, 177, 186, 190
Prototyper 195
Prototyping 50, 54, 92, 94, 112, 115, 122, 126–130, 132, 158, 161f., 190
evolutionäres 126
horizontales 127
Strategie 126
vertikales 126
werkzeug 86, 116, 128, 130, 161
auswahl 129
Prozessgestaltung 53, 81
Prozessmodellierung 85
Prozessmoderation 85
Pseudofunktionalität 161
Pseudopartizipation 29

Q

Quadranten 84, 90, 93
Quadrantenmodell 81, 89, 92
Synchronisation 91
und Projekttyp 90
Qualifizierung 104
Qualitätssicherung 64, 75, 77, 81, 172, 175, 193
quick and dirty 130
Quick Reference 164

R

Rationalisierung 152
Realisierung 96
Realisierungsphase 87
Regelung 82
Registrierkarte 39
Reportgenerator 88
repository 88
Repräsentativität 41, 46, 98, 151
der Beteiligung 40
Ressourcen
notwendige 71
Review 94
RFA-Netze 86
Richtlinien 25
softwareergonomische 31

S

SAA 195
SADT 125
SAM 194
Schädigungsfreiheit 5
Schneeballprinzip 144f.
Schnittstelle
Arbeitsaufgabe 11
und Abstraktion 87
und Erwartungskonformität 132
und EVADIS 132
und Richtlinien 25
und Softwareergonomie 18
zu anderen Anwendungen 153
zu anderen Systemen 44
zwischen strategischer und operativer Ebene 72
Schnittstellen
-gestaltung 25, 99, 161, 167
-güte 132

interaktive 12
-probleme 177
zu viele 181
Schulung 36, 41, 116, 141–146, 168, 178, 189
und Akzeptanz 178
Schulungseinheiten 178
Siemens Nixdorf 130, 196
Simulation 20, 92, 177
Simulations
-methoden 122f., 125
-studie 87, 123, 125
Software VII, 20
-anforderungen 90
-benutzungsangemessen 63
-benutzungsorientierte 80
-entwickler VII, 2, 27, 30f., 57, 75, 110, 193
und Workshops 102
-entwicklung VII, 1, 3, 4, 6, 27, 40, 48, 50, 56, 58, 73, 80, 110, 185
benutzerorientiert 84
optimale 86
Optimalitätskriterien 83
partizipative 81
technische Pannen 97
und Beteiligungsmodell 97
und Optimierungszyklen 83
und Projekttyp 88
und Regelung 83
und Steuerung 83
und Tests 83
-entwicklungs
-abteilung 93
-konzept 89
-praxis 55
-projekt 1, 4, 90, 95, 97
-prozess 52, 76, 97
-team 98
-ergonomie 1, 3, 17, 37, 112
-fehlerhafte 39
-firma 59, 62
extern 56, 66
-flexibel einsetzbar 111
-gestaltung 93, 198, 211
-interaktive 16
-kosten 178
-lieferant 56
-lösung 62, 98
-produkt 131f.
-projekt 51, 53f., 57f., 60, 73
Aufbauorganisation 61
Erfahrungen 74
-system 90, 104
-und Anpassung 31
-und Arbeitsaufgabe 111
-und Aufgabenangemessenheit 27
-und Hypertext 194
-Weiterentwicklung 64
-werkzeug 194
Anwendung 97
Haus 39, 93
Lebenszyklus 75, 77
und Arbeitsabläufe 1
soziotechnisches System 7
Spezifikation 90, 96
Spezifikations
-methoden 86
-phase 90
Standardsoftware 3, 39, 111, 168, 170f.
Steuerung 82
storyboards 125
Styleguide 130
SUMI 196
Support 39, 168
-stelle 147
Synchronisation
divisionale 92
informationelle 93
personelle 93
System
-abstürze 180
-analyse 64
-änderungen 164
-anpassungen 177
-Auswahl 177
-bedienung 178
-benutzern 179
-benutzung 189
und Arbeitshilfen 189
-beschreibung 186
Beurteilung 164
-einführung 64, 179–181, 189
-entwicklung
evolutionäre 82
-flexibilität 179
-Funktionalität 163, 184
-gestaltung 8, 9, 14, 132, 162
-konzeption 64
-lieferant 175
-performanz 180
-potentiale 164
-soziotechnisches 8, 71, 182
-spezifikation 64, 178
-technisches 165, 175, 179–181, 185, 189
-test 64
-theorie 82
Tuning 180
Umgang mit 165
und aufgabenbezogene Unterstützung 179
und Denkfehler 180

und PPS 177, 179, 181
-unterstützung 164
-version, aktuelle 177
Szenarium 92, 124

T
TASK 113
technische
Durchführbarkeit 113
Grenzen 105
Lösungen 111
Möglichkeiten 105, 111
Unterstützung 113
Test 41f., 83, 87, 90–92, 94, 133, 136, 159, 167, 184
Aufgabe 135, 141
Benutzergruppe 136
-daten 90, 133, 190
-durchführung 190
-ergebnisse 32, 160
-leiter 87f., 133f., 136, 138f.
Hilfestellung 138
lautes Denken 137
logfile recording 135
mit Benutzern 43
-personen 136
Anzahl 136
-phase 85, 91
-prozedur 138
-serie 138
-system 135, 177
-zwecke 177
Test-Aktivitäts-Zyklus 84
Toolbook 195
Transparenz 15–17, 68, 95, 98, 106, 152, 201
Beispiel 16
und Gerüchte 187

U
Übung 143
praktische 142
Übungs
-arbeitsplatz 143f., 189
-beispiele 137
-möglichkeit 142
UIMX 195
Umfrage 39, 115, 121, 166, 170, 188
-ergebnisse 116
Unterstützung 15f.
Usability
Test 92f.
Usability Lab 3
User-Group 39, 166

V
VDI-5005 15f.
Versionenkonzept 90
zyklisches 84
Versioning 92
Versionshistorie 172
Video
-aufnahmen 88, 94
-konfrontation 138
VisualBasic 195

W
Walk-through 94
Warnsignale 74
Wartung 81
Wartungsaufwand 90
Wartungsprobleme 82
Werkzeug 11, 41, 53f., 74, 82, 87f., 111, 116, 124f., 127, 161f., 184, 190, 192
benutzerorientiert 127
schnittstelle 13, 86
zur Entwurfsanalyse 88
WindowsMaker 195
wizard of oz 92, 123
Workshop 94, 123, 165

Y
YOURDAN 125

Z
Zeitpunkt der Beteiligung 42
Zumutbarkeit 5, 7
Zusammenarbeit VII, 2, 31f., 39, 44, 79, 96, 115, 163, 166f.
Auftraggeber 60
bei der Systementwicklung 161
Bereitschaft zur 2
direkte 115
fruchtbare 47
hilfreiche Voraussetzungen 37
in Workshops 159
Koordination 26
mit Spezialisten 37
und Ausbildung 185
und Bedenken 156
und Entäuschung 185
und Konflikte 186
und Motivation 118
und Prototyping 127
und Qualifizierung 38
von EDV-Fachleuten 125
Voraussetzungen 57
Zyklusdauer 91f.
Zyklusdurchläufe 91

Anschrift der Autoren:

Dipl.Inform., Dipl. Psych. Matthias Rauterberg,
Dr. Philipp Spinas,
Dipl.Psych. Oliver Strohm,
Prof. Dr. Eberhard Ulich
Institut für Arbeitspsychologie (IfAP)
Eidgenössische Technische Hochschule (ETH)
Nelkenstrasse 11, CH-8092 Zürich

lic.phil. Daniel Waeber
CIM CBT, MTO-Team
c/o HWV Olten
Bifangstrasse 10, CH-4600 Olten